

205  
AN INVESTIGATION OF TRAFFIC CONGESTION IN AN AUTOMATED GUIDED  
VEHICLE MATERIAL HANDLING SYSTEM USING ANIMATION (CINEMA/EGA)

by

INGKO OETOMO

B.S., KANSAS STATE UNIVERSITY, 1986

-----  
A MASTER'S THESIS

submitted in partial fulfillment of the

requirements for the degree

MASTER OF SCIENCE

Department of Industrial Engineering

KANSAS STATE UNIVERSITY  
Manhattan, Kansas

1988

Approved by:

*P. E. Thosh*

-----  
Major Professor

LD  
2668  
IT4  
IE  
1988  
037  
C. 2

A11208 130339

## TABLE OF CONTENTS

	Page
LIST OF FIGURES .....	i
LIST OF TABLES .....	ii
ACKNOWLEDGEMENTS .....	iii
INTRODUCTION .....	1
PROBLEM AND OBJECTIVES .....	1
GENERAL DESCRIPTIONS OF AGVS .....	4
SIMAN/CINEMA .....	8
PROGRAMMING APPROACHES .....	9
MODELING .....	11
CASE 1 .....	16
CASE 2 .....	36
CONCLUSION .....	52
BIBLIOGRAPHY .....	58
APPENDIX A: PROGRAM FOR CASE 1 .....	60
APPENDIX B: PROGRAM FOR CASE 2 .....	100

# LIST OF FIGURES

Figure	Description	Page
1	AGV Symbol .....	15
2	Case 1 - Basic Layout .....	18
3	Case 1 - Throughput (Basic Layout) .....	26
4	Case 1 - Improved Layout .....	30
5	Case 1 - Throughput (Improved Layout) .....	32
6	Case 2 - Basic Layout .....	38
7	Case 2 - Alternative 1 .....	39
8	Case 2 - Alternative 2 .....	40
9	Case 2 - Alternative 3 .....	41
10	Case 2 - Throughput (Basic Layout) .....	46
11	Case 2 - Improved Layout .....	48
12	Case 2 - Throughput (Improved Layout) .....	51
13	Cinema Symbols for Station and Queue .....	53
14	Problems with Station Symbols in Defining the Route .....	53
15	Actual vs Preferred Display of Parts in Queues ...	55

# LIST OF TABLES

Table	Description	Page
1	Case 1 - From-To-Chart .....	17
2	Case 1 - Total Loadfeet .....	19
3	Case 1 - Maximum Queue Size Algorithm (Basic Layout) ..	24
4	Case 1 - Shortest Distance Algorithm (Basic Layout) ..	24
5	Case 1 - Parts Waiting Time (Basic Layout) .....	28
6	Case 1 - Maximum Queue Size Algorithm (Improved Layout) .....	31
7	Case 1 - Shortest Distance Algorithm (Improved Layout) .....	31
8	Case 1 - Parts Waiting Time (Improved Layout) .....	34
9	Case 2 - From-To-Chart .....	37
10	Case 2 - Total Loadfeet .....	42
11	Case 2 - Maximum Queue Size Algorithm (Basic Layout) ..	44
12	Case 2 - Shortest Distance Algorithm (Basic Layout) ..	44
13	Case 2 - Parts Waiting Time (Basic Layout) .....	45
14	Case 2 - Maximum Queue Size Algorithm (Improved Layout) .....	49
15	Case 2 - Shortest Distance Algorithm (Improved Layout) .....	49
16	Case 2 - Parts Waiting Time (Improved Layout) .....	50

## ACKNOWLEDGEMENTS

I would like to thank my major advisor Dr. L. E. Grosh for his guidance and invaluable advice in the course of this study.

To my parents who always have faith in me. To Dr. S. A. Konz and Dr. Prakash Krishnaswami for serving on my committee. And to Dr. B. A. Kramer for his continuous encouragement.

## INTRODUCTION

To investigate traffic congestion in an AGV Material Handling System one needs a topic area and a simulation language. We followed the work of Read (A Study Of Implementation And Evaluation Techniques Of Advanced Guided Vehicle Systems) to furnish the AGV system. The original work was done using GPSS/H on a mainframe computer. Since that time, System Modeling Corp. has marketed SIMAN/CINEMA - a computer simulation and animation package. A professor's package, an unrestricted version, was purchased and used for the research presented in this thesis.

### Problem and objective.

There are a number of researchers who have addressed AGV systems. Egbelu and Tanchoco [2] developed a simulation package to evaluate different rules for priority assignment at intersection, vehicle selection, and job selection. Their results indicated that dispatching rules based on a measure of distance might perform poorly if the layout of the facility and locations of the stations were not carefully designed.

Russell [11] evaluated different vehicle dispatching rules and their effect on shop performance. She used a Q-GERT simulation model and employed one computer dispatched lift-truck. Her results showed that different vehicle dispatching rules did not affect shop performance

significantly, however, they affected the length of the queue.

Stauffer [13] addressed the strategy to manage the routing of AGVs carrying empty racks in an AGV system (AGVS) which was integrated with an Automatic Storage and Retrieval system (AS/RS), using a 3-D graphical animation. He found that both the continuous check and time delay check to find the next job were not very effective in his proposed AGVS.

In 1985, Norman [7] developed a simulation package for automated guided vehicle as an end result of her master thesis. She used SIMAN to develop the package and employed the shortest k-path algorithm to route the AGVs.

Read [10], in 1985, developed a procedural technique in designing AGV systems. He suggested four procedural steps:

1. Loadfeet directioning: a pre-simulation procedure to determine the direction of the traffic flow.
2. Vehicle estimation: a pre-simulation procedure which determines the range number of AGVs needed in an AGV system based on the calculated loadfeet obtained in (1).
3. Cutoff implementation: a process to determine where possible cutoffs or shortcuts are needed to shorten travel distance between input and output stations.
4. Block division: a process to determine where a smaller control segment is needed to reduce the amount of congestion time in the system.

Most of researches on AGV systems have addressed or referred indirectly to the problem of traffic congestion but only a few have dealt directly through some sort of

statistical measurements. The purpose of this thesis is to investigate traffic congestion in an automated guided vehicle material handling systems using animation (CINEMA/EGA)



## GENERAL DESCRIPTIONS OF AGVS

The Material Handling Institute defined an AGV as follows:

"a vehicle equipped with automatic guidance equipment, either electromagnetic or optical. Such a vehicle is capable of following prescribed guide paths and may be equipped for vehicle programming and stop selection, blocking and any other special functions required by the system".

There are basically five types of vehicle that are used in AGVSs: automated towing vehicles, pallet trucks, unit load carriers, automatic fork lift trucks, and specialty trucks (sideloading trucks, for example). These vehicles are powered by lead-acid batteries, with capacities of 24 to 48 volt and 80 to 500 amp/hour, and can remain charged for eight to 16 hours.

There are three different AGV/controller communication methods that can be used in an AGVS:

1. Through guidewire - guidewire transfers information on AGV location and status to system controller. This method minimizes chance of signal interference, but AGV must be over the guidewire to communicate to the controller.
2. FM radio link - Fm radio signals which contain information on AGV location and status, are transmitted between AGV and system controller. Communication can occur at long range, however, federal approval of the FM broadcast frequency is required, and other FM

signals can interfere with communication particularly in urban environments.

3. Optical - light sensor tracks optical or chemical path. Layout is easy to change, but clear and clean floors are needed.

According to Kulwiec [4], there are two types of control for AGV systems: traffic management and system management. Traffic management prevents vehicles collisions and determines the path the vehicle should travel.

Traffic management can be grouped in two types: zone (block) control and sensor control. Zone (block) control breaks the system into several zones and allows only one vehicle in a zone at one time. Only after the vehicle leaves the zone can another vehicle enter the zone.

Sensor control employs optical, physical contact and/or sonar sensors to avoid collision. Generally, a specified distance between vehicles is maintained.

The other type of control, system management, determines which task is to be performed. It can be done in three ways or any combination of them:

1. On-board dispatching - operator dispatches the vehicle to the desired station by entering the appropriate codes to the panel on board the vehicle.
2. Remote terminal dispatching - operator dispatches the vehicle to the desired station from a remote terminal.

3. Central computer dispatching - computer controls and monitors all vehicle in the system. Usually this computer is also interfaced with other components of production (e.g. AS/RS).

An AGV operates in various areas. Bischoff [1] categorizes the traditional application areas of AGVS into three:

1. warehouse/distribution applications,
2. production applications, and
3. institutional applications such as offices and hospital.

AGVs can perform a wide range of function in these area of applications. Some function of AGVs are :

1. supplying and collecting material;
2. order picking;
3. production integrated application (e.g. transporting assembly through a set of operation);
4. interfacing with an AS/RS.

New areas of application of AGV have emerged as well as new functions. The marriage of an AGV and a robot has given the AGV a new function as a mobile workstation. In airports, an AGV could work as a "people mover".

There are various benefits of AGVS, depending on their area of application. Some of the benefits are listed by Kulwiec as follow;

1. Automatic Interfacing - the interface between vehicle and pickup/delivery points is unmanned.
2. Easily adapted to automation - Vehicles can be easily interfaced with automatic equipment.
3. Tighter Material Control - the use of computer control system allows monitoring of exact location of all

material in transit.

4. Increased productivity - through reduction in labor.
5. Economic Justification - Appropriate application should yield a return on investment of three years.
6. Flexible System Capacity - Vehicle can be removed or added from the system depending on the amount of activity.
7. Flexible Use of Vehicle - New or different assignment can be easily programmed.
8. Ease of Installation - Easily adapted to existing facility with little interruption of daily operation.
9. Easy Modification and Expansion of The Track Layout - minimum interruption to daily operation.
10. Efficient Use of Floor Space - Require less floor space than that required by floor-installed conveyors.

New application of AGVs in multiple, identical assembly lines offers the benefit of reducing the effect of individual failure in any lines. In this type of assembly lines, the AGVs transfer the parts from failed stations to the operational stations of other lines.

## SIMAN/CINEMA

SIMAN (SIMulation ANalysis) is a simulation language developed by C. Dennis Pegden and supported by Systems Modeling, Corp. SIMAN is a FORTRAN based simulation language, implemented for mainframe, mini, and personal computers.

SIMAN can be used in the three modeling orientation -- process, event, and continuous, as well as any combination of these orientations. The process orientation is used to model discrete change systems and uses a block diagram to construct the flow of entities through the system.

The event orientation, used to model discrete change systems, relies on user-defined FORTRAN subroutines to describe the event logic. The user-defined subroutines are mainly used to augment the options not available in the block diagram. However, they can be used to replace the block diagram components.

The continuous orientation is used to model processes that can be represented by a set of algebraic, differential, or difference equations. These continuous components are coded in FORTRAN within the subroutine STATE.

SIMAN modeling framework stresses the distinction between the system model and the experimental model. The system model describes the feature or logic of the model, while the experimental model defines the parameter values

used in the system model. This type of framework allows various simulation experiments by altering values specified in the experimental frame.

CINEMA is a computer animation system based on the SIMAN simulation language. It's developed by System Modeling, Corp. CINEMA animation construction involves building a SIMAN model to represent the system being animated and the construction of the animation layout using CINEMA.

CINEMA layout consists of two components - the static and the dynamic model. The static component or the static background consists of objects that do not change sizes, locations, colors or shapes during the simulation (for example: walls, aisles). The dynamic component consists of objects that change sizes, locations, colors or shapes such as parts, machines, robots, workers and so forth.

#### Programming Approaches

There are two ways AGVs can be represented in SIMAN: transporters or entities. Transporters are those material handling devices that move items one load unit at a time along a fixed or varied path. Entities are those objects that engage in activities. For examples, workpieces in a manufacturing facility, or customers in a service facility would be considered entities.

Modeling AGVs as transporters allows the utilization of the SIMAN transporter features such as vehicle allocation,

vehicle selection rules. However, early trials showed that the AGV symbols disappeared at queues during the animation. Additionally, this method requires detailed distance specification between station pairs in the experimental model. Therefore the entity method of representing AGVs is considered.

The entity representation of the AGV produces better animations. Though this method does not enable us to use all the transporter features, we can imitate the above features with some extra programming. Some of the SIMAN entity features that could be used to model AGVs system are sequence (route) and some task selection rules. This method also requires less detailed distance specification between station pairs.

Loads or jobs can be represented in a couple of ways. Read used the capacity of a storage to represent the loads. When a load was created, the load was sent to the storage and the storage capacity was incremented. When the load was picked up, the storage capacity was decremented.

This method worked well in the simulation. However, it produced a poor animation. In CINEMA a storage is defined as a resource. A resource has two different states: busy and idle. When the loads are represented as the capacities of a storage, only two different displays can be shown: the presence of a load or loads, and the non existence of the load. The number of loads cannot be displayed.

Another method is to represent the loads using entities. When a load was created, the load was sent to the station and placed in a queue at the station to wait for the allocation of an AGV. This method enabled us to display the number of loads at a station. Additionally, this method would allow us to display a load which has been allocated an AGV and is waiting until the AGV can pick it up (this cannot be done when loads are represented by the capacities of a storage).

### Modeling

To accurately simulate the AGVSS, the simulation programming includes blocking, routing, parking, dispatch and scheduling algorithm.

Blocking is accomplished using the zone control method. The guide path is divided into small segments. Each segment is considered as a zone and only one vehicle is permitted in a segment at one time. If a vehicle wants to enter a segment which is being occupied by another vehicle, it has to wait until the other vehicle leaves the segment.

In SIMAN blocking is achieved by taking the segments as a resource. A resource is a storage which can have single or multiple capacities (units). An entity (AGV) can allocate one or several units of the resource by seizing the resource. When the resource unit(s) is not available at the time it is allocated, the entity will have to wait in a queue. To simulate the zone control, the capacity of the resource is



limited to one.

Routing is choosing the shortest possible route to the vehicle destination. Routing is accomplished using the SIMAN sequence feature. The sequence feature defines a station visitations sequence that the AGVs will visit. The distance between stations is considered as a segment.

Two vehicle dispatching and scheduling algorithms were considered: the maximum queue size and the shortest distance. The maximum queue size algorithm checks all input stations and selects the one with the most requests. This method is done in the process orientation using a block diagram.

The shortest distance algorithm chooses the closest input station which has a request or requests for the AGVs. This algorithm is implemented in the event orientation using FORTRAN subroutines. The two algorithms were chosen to explore SIMAN process orientation and event orientation.

A parking station was provided to place empty AGVs. Empty AGVs without any request can circulate in a loop or be placed in their last drop-off station. However, these two methods will obstruct other AGVs with jobs to do.

The CINEMA background for the AGV system consists of

1. walls, aisles and intersections,
2. locations of pick-up and drop-off stations,
3. locations of queues, and stations needed to define the segments,
4. locations of loading/unloading place at each pick-up or drop-off station.

The first two items are displayed in the animation but the last two are not. The locations of queues are needed to place the symbol of AGVs waiting to seize the segments and to display the symbol for loads waiting for the allocation of AGVs or waiting for the arrival of the allocated AGVs. The stations (3) are needed to define the segments and the path the AGVs will travel to and through a segment.

The loading/unloading locations at each input or drop-off station need to be defined to display loading/unloading AGVs. If the location is not defined, an AGV undergoing a time delay (loading/unloading time), will disappear from the screen and will reappear again when it finishes the time delay.

The AGV is the only dynamic component of the CINEMA layout. However, to make the observation of the animation more effective, the job is also considered as one of the dynamic components. The symbol for a job waiting to be allocated an AGV is made different from the symbol of a job waiting for an AGV to arrive.

There are three different AGV states that are animated: loaded, unloaded but allocated to a job, and unloaded without a request. The need for two different unloaded AGV states is deemed important in interpreting the cause of congestions, and to observe the unloaded AGVs. Several symbols are used to display the loaded AGV. The different symbols are used to

tell the origin of the job and the destination of the job being carried on the AGV (see Figure 1).




SYMBOL	COLOR	DESCRIPTION
	Yellow or green	Empty without a request 1. parked in parking lot 2. going to parking lot
	Blue	Going to pick up a part
	25 (36) color combinations (i.e. 5 (6) on the left, and 5 (6) on the right)	Transporting a part Color: Left side: station of origin Right side: destination

FIGURE 1. AGV SYMBOLS

## CASE 1

The first facility where the AGVS was studied had 18 input stations and 16 drop-off stations. The data for the number of jobs at each stations and their destination per shift was shown in a "from-to" chart in Table 1.

In the beginning, the basic layout (guidepath) for the AGVS was constructed (see Figure 2). The basic layout was constructed by connecting decision points (input stations, drop-off stations, and turnoffs) with a straight line.

After completing the guidepath, the traffic flow was determined using the loadfeet directioning method. In the loadfeet directioning method, the guidepath is divided into separate sections between decision points. The flow of loads is calculated across each section. The flow of loads for each section is multiplied by its section length and summed over the whole system for each particular traffic pattern. The result is in loadfeet units. The sums for each of the traffic pattern are compared and the one with the smallest value is adopted as the traffic flow.

Two traffic flows were considered: clockwise and counter clockwise flows. Table 2 shows the calculated total loadfeet. The total loadfeet for the clockwise flow was smaller than that of the counter clockwise flow, and was chosen as the traffic flow.

TABLE 1. CASE 1 - FROM-TO-CHART

INPUT	OUTPUT																	
	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34		
1			.8		.2		.4	.4	.4		.7	.1					3.0	
2	3.6			1.0	5.6												10.2	
3	16.2	.4		.2			.2	2.8	.2								20.0	
4	33.8	.2															34.0	
5	10.7										.1						10.8	
6	9.6																9.6	
7	10.7										.1						10.8	
8	5.2	.4	1.4	10.4	8.0			1.0			5.0	.6	12.4		1.4		45.8	
9	4.8		2.0	.6	1.6			2.1			2.8	.4	1.2		11.1	1.0	27.6	
10	.4		.2	2.2		2.4					.7	.1			4.8	.2	11.0	
11				12.8	3.8						1.2	.1	3.8				21.7	
12	2.8	.6		2.2	3.2			.9			1.3	.2	2.2		.2		13.6	
13	4.6		.4	1.4	1.2	.2	.8	1.2		3.3			.2		.6		13.9	
14	45.2		.8	.8	9.4			.2		.4							56.8	
15	2.8																2.8	
16		5.2	16.4	24.8	21.1	.4	4.0	5.6	1.8	1.0	12.4	1.8	1.6	2.9	5.8	.2	105.0	
17	7.8			.4			4.4	.2		3.9	.7	.1	.2				17.7	
18	.2						.2		.2	.2	.2						1.0	
	158.4	6.8	22.0	56.8	54.1	3.0	10.0	14.4	2.6	8.8	25.2	3.4	21.6	2.9	23.9	1.4	415.3	

THROUGHPUT :

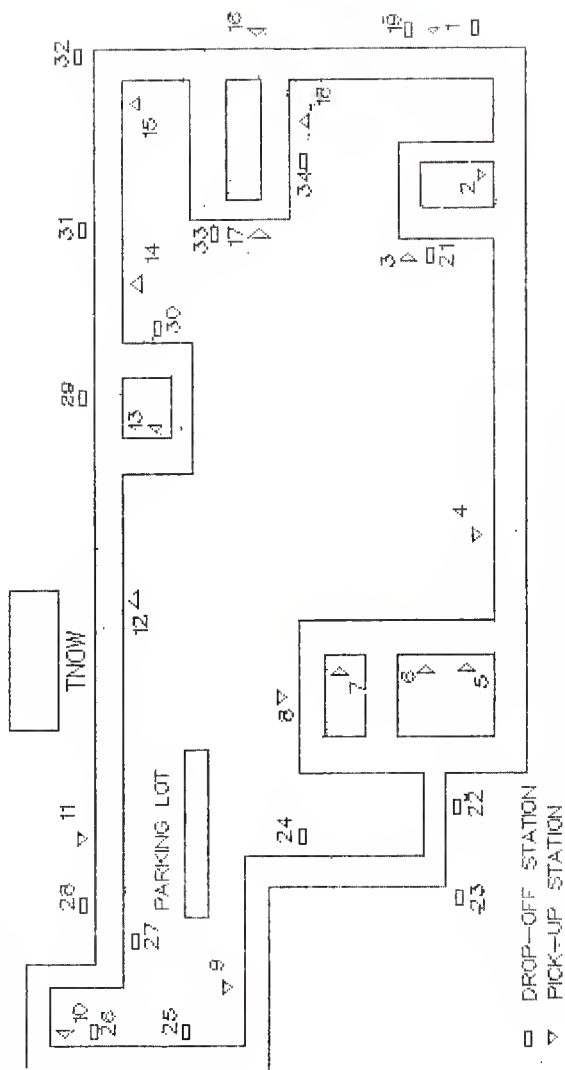


FIGURE 2. CASE 1 - BASIC LAYOUT

TABLE 2. CASE 1 - TOTAL LOADFEET

BLOCK	DISTANCE	NUMBER OF LOADS		LOADFEET	
		CW	CCW	CW	CCW
1 -- 19	12	170.7	250.6	2048.4	3007.2
1 -- 20	24	173.7	247.6	4168.8	5942.4
2 -- 35	6	144.9	234.4	869.4	1406.4
2 -- 36	18	155.1	224.2	2791.8	4035.6
3 -- 21	10	42.0	0.0	420.0	0.0
3 -- 35	96	22.0	20.0	2112.0	1920.0
4 -- 36	152	175.1	246.2	26615.2	37422.4
4 -- 37	68	209.1	212.2	14218.8	14429.6
5 -- 6	30	10.8	66.2	324.0	1986.0
5 -- 37	20	0.0	77.0	0.0	1540.0
6 -- 7	32	0.0	56.6	0.0	1811.2
6 -- 38	98	20.4	0.0	1999.2	0.0
7 -- 8	30	10.8	45.8	324.0	1374.0
8 -- 38	98	56.6	0.0	5546.8	0.0
9 -- 24	116	172.2	249.1	19975.2	28895.6
9 -- 25	102	199.8	221.5	20379.6	22593.0
10 -- 26	24	175.4	245.9	4209.6	5901.6
10 -- 27	104	186.4	234.9	19385.6	24429.6
11 -- 12	192	196.7	224.6	37766.4	43123.2
11 -- 28	32	175.0	246.3	5600.0	7881.6
12 -- 42	82	210.3	211.0	17244.6	17302.0
13 -- 30	56	17.3	0.0	968.8	0.0
13 -- 42	44	3.4	13.9	149.6	611.6
14 -- 31	40	252.4	168.9	10096.0	6756.0
14 -- 43	46	195.6	225.7	8997.6	10382.2
15 -- 31	94	230.8	190.5	21695.2	17907.0
15 -- 32	12	233.6	187.7	2803.2	2252.4
16 -- 44	36	205.4	171.9	7394.4	6188.4
16 -- 45	6	310.4	66.9	1862.4	401.4
17 -- 33	22	1.4	42.6	30.8	937.2
17 -- 34	56	19.1	24.9	1069.6	1394.4
18 -- 34	36	17.7	26.3	637.2	946.8
18 -- 45	12	18.7	25.3	224.4	303.6
19 -- 45	66	329.1	92.2	21720.6	6085.2
20 -- 35	92	166.9	254.4	15354.8	23404.8
21 -- 36	44	20.0	22.0	880.0	968.0
22 -- 23	50	229.3	192.0	11465.0	9600.0
22 -- 38	22	286.1	135.2	6294.2	2974.4
23 -- 24	60	175.2	246.1	10512.0	14766.0
25 -- 26	62	189.8	231.5	11767.6	14353.0



27 -- 28	4	183.8	237.5	735.2	950.0
29 -- 42	30	206.9	197.1	6207.0	5913.0
29 -- 43	6	181.7	222.3	1090.2	1333.8
30 -- 43	30	13.9	3.4	417.0	102.0
32 -- 44	78	230.7	190.6	17994.6	14866.8
33 -- 44	96	25.3	18.7	2428.8	1795.2
37 -- 38	114	209.1	135.2	23837.4	15412.8

Total loadfeet	372633.0	385607.4
----------------	----------	----------

(CW = Clockwise

CCW = Counter clockwise)

The next step was to calculate the range number of AGVs needed to transport all loads. First, the vehicle travel second per shift was calculated as follows:

$$\frac{\text{vehicle travel}}{\text{seconds per shift}} = \frac{\text{total load-feet distance}}{\text{vehicle speed}}$$

Then, the total loading/unloading time was added to find the vehicle work seconds per shift.

$$\frac{\text{vehicle work}}{\text{second per shift}} = \frac{\text{vehicle travel}}{\text{seconds}} + \frac{\# \text{ jobs}}{\text{per shift}} \times \frac{\text{load/unload}}{\text{seconds}}$$

The range number of AGVs needed was calculated by converting this value into vehicle work per shift and dividing it with "traffic congestion factor" and "idle time factor". Kulwiec [4] suggested a traffic congestion factor value of 0.85 , and an idle time factor value between 0.6 and 0.8. However, Read [10] thought 0.8 was a very conservative estimate for traffic congestion and suggests using 0.95. Read also suggested an idle time factor value between 0.5 and 0.8.

$$\begin{aligned} \text{vehicle work} &= \frac{\text{vehicle work second per shift}}{(60 \text{ s/min}) * (60 \text{ min/hr}) * (8 \text{ hr/shift})} \\ \text{range \# of vehicles} &= \frac{\text{vehicle work}}{\text{traffic congestion factor} * \text{idle time factor}} \end{aligned}$$

Adopting Read's suggestion, the range number of the AGVs was calculated as follows:

$$\begin{aligned}\text{Vehicle work second per shift} &= (372633/4) + (412.5*90) \\ &= 1320283.25\end{aligned}$$

$$\text{Vehicle work} = 1320283.25 / (60*60*8) = 4.77$$

$$\text{Upper range number of the AGVs} = 4.77 / (.95*.5) = 10$$

$$\text{Lower range number of the AGVs} = 4.77 / (.95*.8) = 6$$

Using the calculated range number, the AGVs were simulated and animated. At the beginning, the AGVs were simulated and animated using the upper range number of AGVs with original guidepath. The original guidepath was the guidepath with the least number of segments. The purpose of this was to see where congestions occurred, and which segments should be shortened to reduce the effect of the congestion.

With the original guidepath, we found that congestion almost always occurred when there is an AGV loading/unloading a job. The segments in the area with frequent and long congestion were then cut into smaller segments.

Using the edited guidepath, the simulation for the AGVs was rerun using the calculated range number of AGVs. In evaluating the AGVs, the animation and the loaded and unloaded travel time, loading/unloading time, parking time, waiting time, and throughput or the number of part transported, were used.

System utilization and system efficiency were also considered to evaluate the AGVS. Hitchen [3] described

system utilization as the percentage of time that all vehicles are transporting a load within the system, and the system efficiency as the percentage of time that all vehicles with a job to do are in motion.

Read refined the system utilization to include vehicle load and unload times. Additionally, he suggested the inclusion of include vehicle waiting time in the system efficiency to give a truer indicator of system congestion. This leads to the following equations for system utilization and system efficiency to

$$\text{system utilization} = \frac{\text{total loaded travel time} + \text{total loading/unloading time}}{\text{total travel time}}$$

$$\text{system efficiency} = \frac{\text{sum of vehicle travel time} - \text{sum of vehicle waiting time}}{\text{sum of vehicle travel time}}$$

However, because different number of AGVs had different throughputs, these two measurements were not considered appropriate to evaluate the AGVs and were not used.

Table 3 and 4 display the average throughput and percent loaded, unloaded travel time, loading/unloading time, waiting and parking time (per vehicle) for both algorithms. From the system throughput, 10 AGVs were needed to satisfy system requirements when the shortest distance algorithm (SDA) was used. When the maximum queue size algorithm (MQSA) was used, more than 10 AGVs were needed.

TABLE 3. CASE 1 - MAXIMUM QUEUE SIZE ALGORITHM (BASIC LAYOUT)

#	AVERAGE LOADED	AVERAGE UNLOADED	AVERAGE LOADING/ UNLOADING	AVERAGE PARKING	AVERAGE WAITING	AVERAGE SYSTEM
AGVs	TRAVEL TIME (%VEHICLE)	TRAVEL TIME (%VEHICLE)	TIME (%VEHICLE)	TIME (%VEHICLE)	TIME (%VEHICLE)	THROUGHPUT
6	34.5	37.8	27.8	0.0	8.8	266
7	36.3	37.3	26.3	0.0	11.8	295
8	36.9	37.7	25.4	0.0	13.4	329
9	37.4	37.9	24.8	0.0	16.2	359
10	38.8	37.2	24.2	0.0	18.3	388

TABLE 4. CASE 1 - SHORTEST DISTANCE ALGORITHM (BASIC LAYOUT)

#	AVERAGE LOADED	AVERAGE UNLOADED	AVERAGE LOADING/ UNLOADING	AVERAGE PARKING	AVERAGE WAITING	AVERAGE SYSTEM
AGVs	TRAVEL TIME (%VEHICLE)	TRAVEL TIME (%VEHICLE)	TIME (%VEHICLE)	TIME (%VEHICLE)	TIME (%VEHICLE)	THROUGHPUT
6	51.3	17.1	31.7	0.0	12.9	304
7	48.9	20.1	30.9	0.0	13.8	340
8	48.7	21.9	29.6	0.0	16.7	383
9	47.2	24.6	28.5	0.0	17.9	414
10	45.8	28.1	26.1	0.0	21.1	421

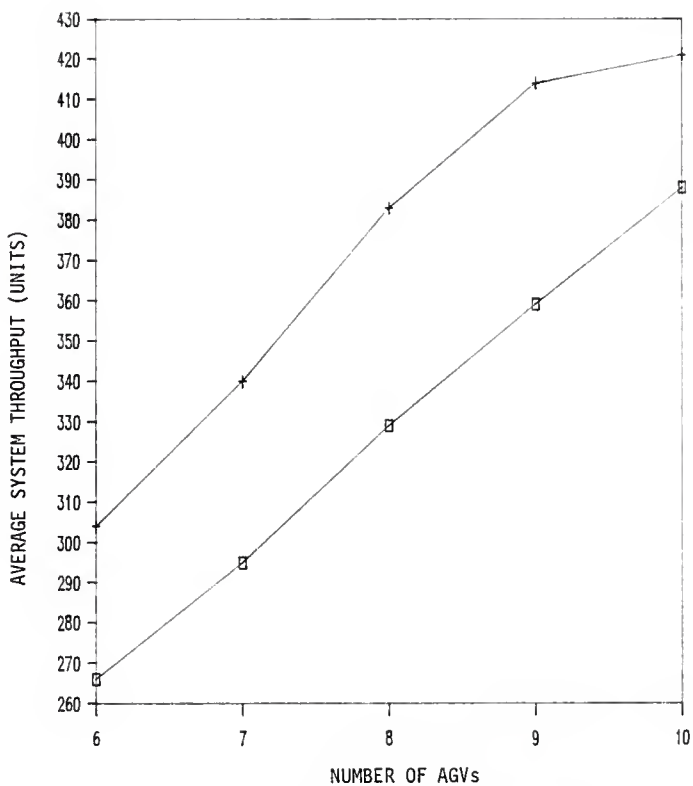
The loaded travel time, unloaded travel time, and loading/unloading time added up to 100 % (Some of the totals did not add up to 100% because of rounding error). The waiting time, the time the AGVs waited to enter a segment, was included in both the loaded travel time and the unloaded travel time.

Figure 3 shows a plot of the average throughputs for both algorithms for different number of AGVs. The SDA transported more parts than the MQSA. This was also reflected in the average of the loaded and unloaded travel time. The rate of change of the throughput for the shortest distance algorithm between 9 to 10 AGVs did not increased as much as that between 6 and 9 AGVs because with 9 AGVs almost all parts were transported. Therefore the addition of one more AGV only transported a few more parts.

The average loaded travel time for the SDA, on the average, was 30 % more than that for the MQSA. The average unloaded travel time for the SDA, on the average, was 40 % less than that for the MQSA.

The average loading/unloading time for the SDA was higher than for the MQSA. However, they were basically equal when the increase of system throughput was considered. This was also true for the waiting time.

The increase in the waiting time could be interpreted as an increase in congestion time. The congestion time could be



LEGEND: + SHORTEST DISTANCE

MAXIMUM QUEUE SIZE

FIGURE 3. CASE 1 - THROUGHPUT (BASIC LAYOUT)

further lowered by reducing the length of the segments. However, this was not done here because of the difficulty in defining smaller segments for the animation.

The parts waiting time for both algorithms is tabulated in Table 5. The parts waiting time is the time the parts were created until the time the parts were picked up. The average, minimum and maximum parts waiting time for the shortest distance algorithm was smaller than that of the maximum queue size.

From Table 4, we noticed that the average loaded travel time (per vehicle) for the SDA decreased as the number of vehicles in the system increased (Although the total loaded travel time actually increased). This phenomenon could be explained by the fact that when the same seed value for the random number generator was used to create the parts, the number of the parts in the system at any point would be the same in different runs. When more vehicles were introduced to this situation, more parts would be transported. As the closer parts were transported, the empty vehicles would look at parts in further stations. This would result in an increase of the average unloaded travel (as can be seen from Table 4), and a decrease in the average loaded travel per vehicle (depending on the location of the next closest station). This phenomenon can also be observed through the animation.



TABLE 5. CASE 1 - PARTS WAITING TIME (BASIC LAYOUT)

# AGVs	AVERAGE WAITING TIME (minute)		MINIMUM WAITING TIME (minute)		MAXIMUM WAITING TIME (minute)	
	MSQA	SDA	MSQA	SDA	MSQA	SDA
6	71.0	34.7	5.7	2.0	391.6	366.5
7	65.6	33.5	5.7	1.6	364.5	317.0
8	54.7	27.6	5.2	1.9	252.0	178.6
9	40.2	15.8	4.3	1.6	165.4	61.2
10	27.5	9.3	2.6	1.5	178.3	55.0

( SDA = Shortest Distance Algorithm  
 MSQA = Maximum Queue Size Algorithm )

This phenomenon is also occurred when the number of AGVs in the system exceeds what the system requires. Since the number of loads is limited (expected loads = 415.3/shift), we can expect that the loaded travel time per vehicle will decrease as we increase the number of AGVs.

Unlike the SDA, the MQSA did not indicate a decrease in the average loaded travel time per vehicle. Additionally, the average unloaded travel time did not change much. This could be expected as the MQSA was affected by size of the queue rather than the location (distance) of the parts.

From the observation of the animation, shortcuts were introduced to the guidepath as shown in Figure 4. The total loadfeet for the improved guidepath was 303091.5. From the basic guidepath, we found that the lower range of the AGVs was too low to satisfy the system requirement. Therefore we reduced the range for the idle time factor to 0.5 - 0.7. Using the new idle time factor, the range number of AGVs was calculated to be 6 - 9. Again, the average throughput, the average percent loaded, unloaded travel time, waiting time, and parking time (per vehicle) were collected , and tabulated as shown in Tables 6 & 7.

The improved layout required at least 10 AGVs when the MQSA was used, and 8 AGVs when the SDA was used. The average system throughput for both algorithms is plotted in Figure 5. Again, we observed a significant decrease in the rate of

TOTAL THROUGHPUT :

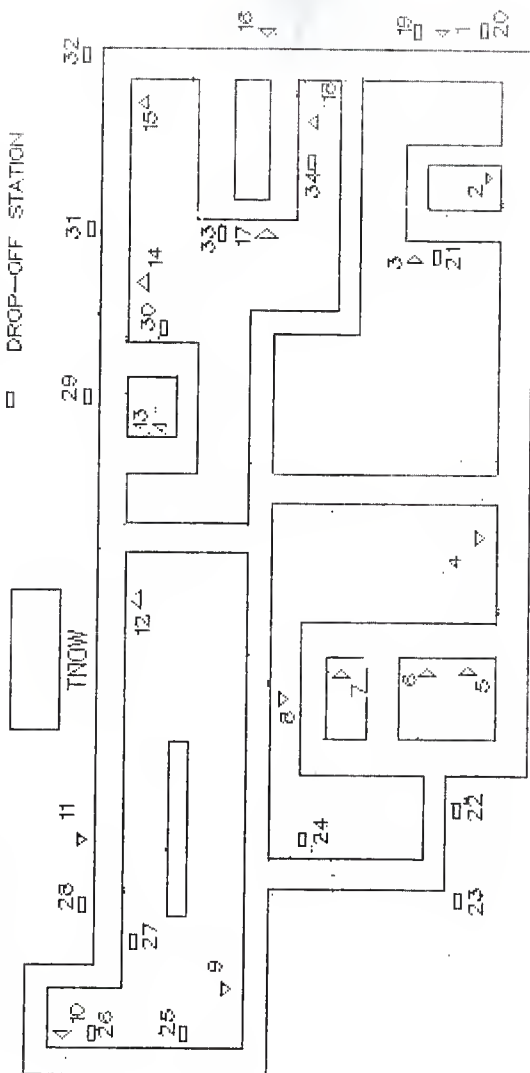


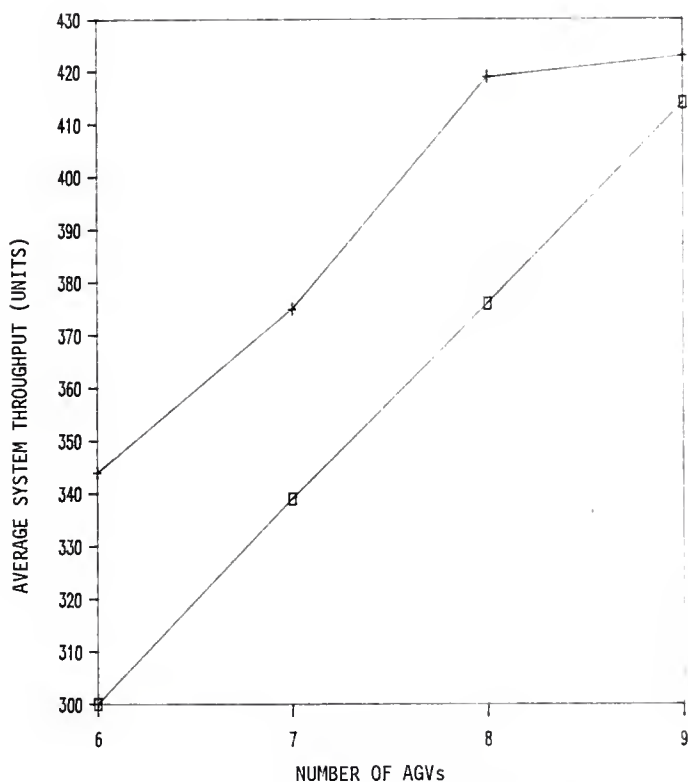
FIGURE 4. CASE 1 - IMPROVED LAYOUT

TABLE 6. CASE 1 - MAXIMUM QUEUE SIZE ALGORITHM (IMPROVED LAYOUT)

# AGVs	AVERAGE LOADED TRAVEL TIME (%VEHICLE)	AVERAGE UNLOADED TRAVEL TIME (%VEHICLE)	AVERAGE LOADING/ UNLOADING TIME (%VEHICLE)	AVERAGE PARKING TIME (%VEHICLE)	AVERAGE WAITING TIME (%VEHICLE)	AVERAGE SYSTEM THROUGHPUT
6	35.3	33.8	30.9	0.0	10.0	300
7	36.1	33.5	30.3	0.0	12.5	339
8	37.1	33.8	29.3	0.0	14.2	376
9	38.6	32.7	28.6	0.0	16.9	414

TABLE 7. CASE 1 - SHORTEST DISTANCE ALGORITHM (IMPROVED LAYOUT)

# AGVs	AVERAGE LOADED TRAVEL TIME (%VEHICLE)	AVERAGE UNLOADED TRAVEL TIME (%VEHICLE)	AVERAGE LOADING/ UNLOADING TIME (%VEHICLE)	AVERAGE PARKING TIME (%VEHICLE)	AVERAGE WAITING TIME (%VEHICLE)	AVERAGE SYSTEM THROUGHPUT
6	44.4	19.9	35.4	0.0	11.0	344
7	44.8	22.1	33.1	0.0	13.2	375
8	42.9	24.7	32.4	0.0	15.8	419
9	40.3	30.2	29.1	0.0	17.6	423



LEGEND: + SHORTEST DISTANCE

MAXIMUM QUEUE SIZE

FIGURE 5. CASE 1 - THROUGHPUT (IMPROVED LAYOUT)

change of the throughput for the SDA between 8 to 9 AGVs because there were fewer parts to be transported.

Table 8 shows the average, minimum and maximum parts waiting time. From Table 8 we noticed that the maximum parts waiting time for the MQSA increased as the number of AGVs reached 9. This indicated that the maximum parts waiting time for the 6 and 7 AGVs did not include the waiting time for the parts that had not been picked up at the end of the simulation (SIMAN data collection method is an activity based method. If the activity with the data to be collected does not occurred at the end of the observation, the data will be lost).

This can also be caused by the congestion and the nature of discrete simulation. As the system throughput increases, we can expect an increase in congestion time. Because the discrete simulation does not allow continuous checking on parts, empty vehicles can check the presence of more request only when it is out of the congestion (queue). While the empty vehicles are waiting in the queue, parts (requests) are being generated at the same time. The generation of new parts will affect which part to be chosen. This will in turn affect the minimum and/or maximum waiting time.

Based on the parts transported only, the average, minimum and maximum parts waiting time for the SDA is smaller than those of the MQSA.

TABLE 8. CASE 1 - PARTS WAITING TIME (IMPROVED LAYOUT)

# AGVs	AVERAGE WAITING TIME (minute)		MINIMUM WAITING TIME (minute)		MAXIMUM WAITING TIME (minute)	
	MSQA	SDA	MSQA	SDA	MSQA	SDA
6	62.7	28.9	5.4	1.9	363.2	346.6
7	50.0	27.5	3.9	1.6	222.5	213.8
8	30.5	12.3	3.7	1.5	171.3	62.0
9	15.9	7.6	1.9	1.3	463.5	46.0

( SDA = Shortest Distance Algorithm  
 MSQA = Maximum Queue Size Algorithm )

Because the utilization rates was gathered to two decimals, no parking time was recorded in the system ( the parking time is very small compared to the simulation run). However, from the animation one could observe that there were a number of vehicles going to or had entered the parking lot before they received more requests. This could be observed by a big increase in the unloaded travel time when the number of AGVs exceeded what the system would require (when there were no more request, the AGVs were sent to parking lot. When a request were issued, the AGV in the parking lot was given the priority to do the job. If there was no AGV in parking lot, then the first empty AGV that spotted the request will do the job).



## CASE 2

The second facility where the AGVS was studied had 24 input stations and 26 drop-off stations. The data for the number of jobs at each stations and their destination per shift was shown in a "from-to" chart in Table 9. Figure 6 displayed the basic guidepath for the second facility.

Three different traffic patterns were considered for this basic layout (see Figure 7, 8, and 9). Table 10 showed the total loadfeet for the three alternatives. Alternative 1 had the least total loadfeet and was chosen as the traffic pattern. The calculated range number of AGVs was 18 to 24.

The required number of AGVs was 23 when the shortest distance algorithm (SDA) was used, and more than 24 when the maximum queue size algorithm (MQSA) was used. The average throughput, and the average percent loaded, unloaded travel time, waiting time, and parking time (per vehicle) were tabulated in Table 11 and 12.

Table 13 displayed the average, minimum and maximum parts waiting time. The average, minimum and maximum parts waiting time for the shortest distance algorithm was significantly smaller than those for the maximum queue size algorithm.

The average throughputs for both algorithms were plotted as shown in Figure 10. Again, the shortest distance algorithm transported more parts than the maximum queue size

TABLE 9 CASE 2 - FROM-TO-CHART

I N P U T	OUTPUT																			
	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44
1					1.0	61.01													9.51	9.51
2			115.01																9.11	9.11
3																			114.41	1.21
4														6.01	6.01					
5					1.6														1.11	1.11
6				115.71															2.21	2.31
7																			2.21	2.21
8																			3.31	3.31
9																			3.31	3.31
10		5.01																	4.41	4.41
11																			120.01	6.61
12																			173.01	2.31
13			3.01			139.41													1.01	1.01
14																			115.01	9.11
15																			116.01	
16																			125.01	
17																			125.01	
18	5.01	4.71		1.0	1.31	1.91	0.61	9.11	9.11	5.21	1.01	2.1111.0111.01	2.01	1.91	2.11				3.11	
19	5.01	4.71		1.0	1.31	1.91	0.61	9.11	9.11	5.21	1.01	2.1111.0111.01	2.01	1.91	2.11				3.11	
20	1.71	2.11		1.1	2.0115.01	4.1	4.1	2.21	2.31	4.1	9.1	2.51	2.51	1.31	9.1	9.1			111.4111.41	111.4111.4111.41
21	5.01	4.71		1.0	1.31	1.91	0.61	9.11	9.11	5.21	1.01	2.1111.0111.01	2.01	1.91	2.11				3.11	
22	5.01	4.71		1.0	1.31	1.91	0.61	9.11	9.11	5.21	1.01	2.1111.0111.01	2.01	1.91	2.11				3.11	
23	5.01	4.71		1.0	1.31	1.91	0.61	9.11	9.11	5.21	1.01	2.1111.0111.01	2.01	1.91	2.11				3.11	
24	5.01	4.71		1.0	1.31	1.91	0.61	9.11	9.11	5.21	1.01	2.1111.0111.01	2.01	1.91	2.11				3.11	
	34.5	10.0	77.4	104.0	5.0	33.5	6.4	79.3	10.1	13.5	40.4	30.3	40.4	40.4	1032.6					

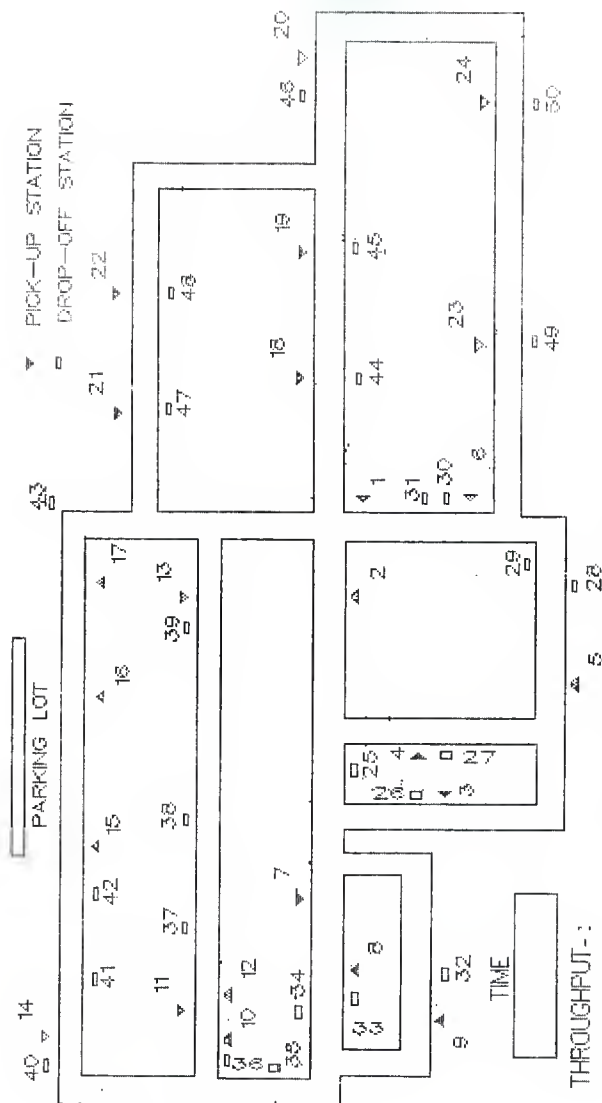
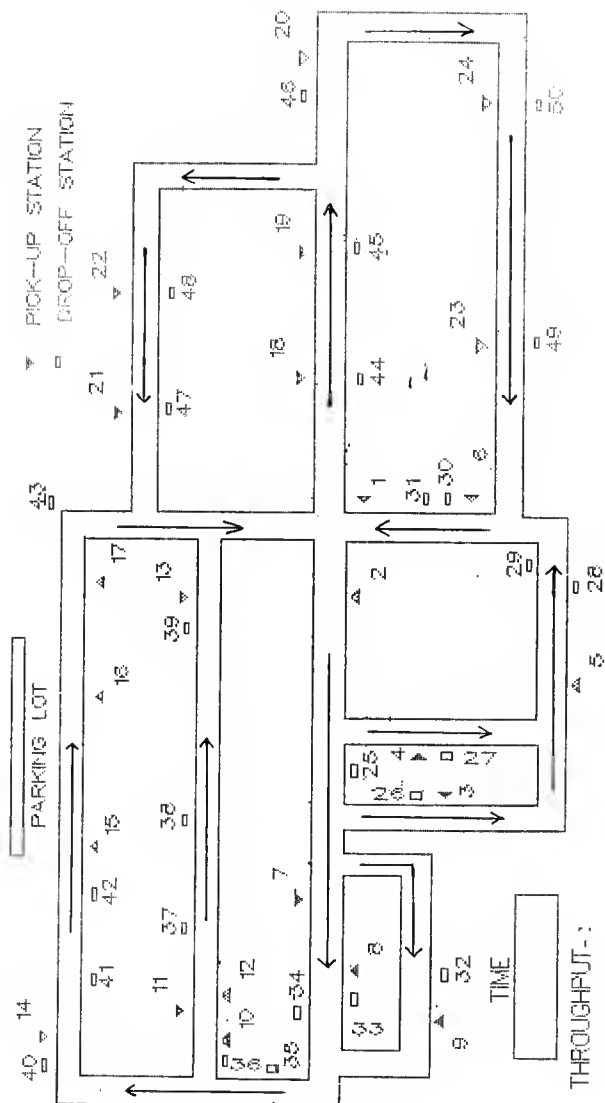


FIGURE 6. CASE 2 - BASIC LAYOUT



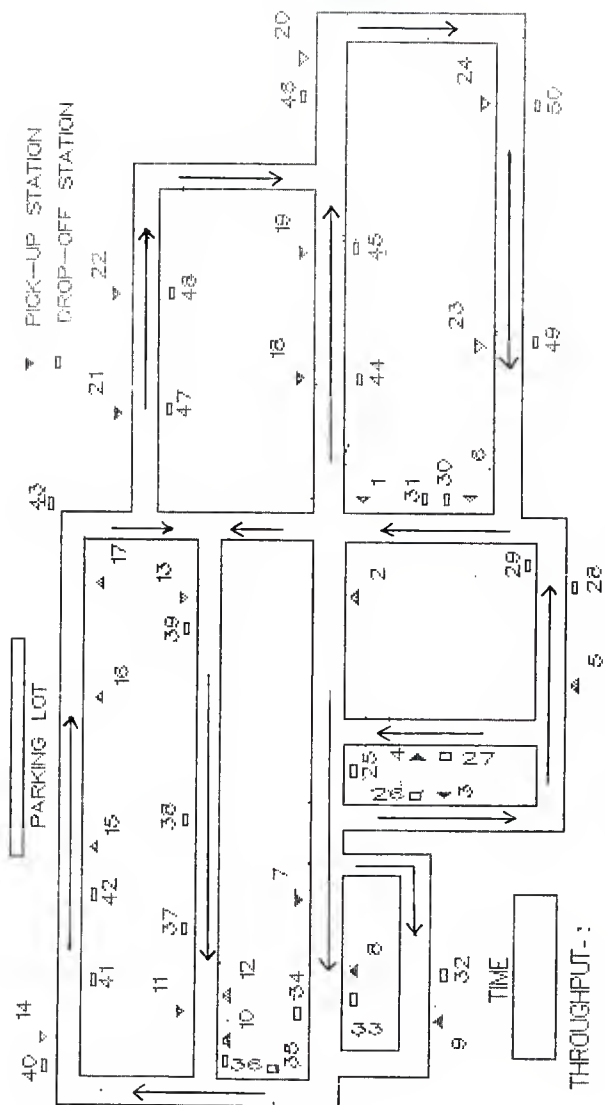


FIGURE 8. CASE 2 - ALTERNATIVE 2



TABLE 10. CASE 2 - TOTAL LOADFEET

BLOCK	FEET	NUMBER OF LOADS			LOADFEET		
		ALT 1	ALT 2	ALT 3	ALT 1	ALT 2	ALT 3
1 -- 31	54.4	404.2	494.0	381.6	21988.5	26873.6	20759.0
1 -- 54	14.7	532.6	622.4	253.2	7829.2	9149.3	3722.0
2 -- 54	62.0	597.4	207.1	394.8	37038.8	12840.2	24477.6
2 -- 57	118.0	666.5	276.2	325.7	78647.0	32591.6	38432.6
3 -- 26	23.5	0.0	149.9	66.3	0.0	3522.7	1558.1
3 -- 59	50.0	22.8	172.7	43.5	1140.0	8635.0	2175.0
4 -- 27	20.6	228.5	16.8	0.0	4707.1	346.1	0.0
4 -- 57	54.4	216.5	28.8	12.0	11777.6	1566.7	652.8
5 -- 28	104.0	240.7	145.3	54.1	25032.8	15111.2	5626.4
5 -- 56	52.0	232.5	137.1	62.3	12090.0	7129.2	3239.6
6 -- 30	8.8	521.2	611.0	264.6	4586.6	5376.8	2328.5
6 -- 55	10.3	491.5	581.3	294.3	5062.5	5987.4	3031.3
7 -- 8	84.0	373.8	78.9	315.7	31399.2	6627.6	26518.8
7 -- 60	36.0	372.4	77.5	317.1	13406.4	2790.0	11415.6
8 -- 33	14.0	375.9	81.0	313.6	5262.6	1134.0	4390.4
9 -- 32	40.0	0.0	0.0	7.9	0.0	0.0	316.0
9 -- 61	113.8	2.1	2.1	5.8	239.0	239.0	660.0
10 -- 11	16.0	185.5	259.5	511.9	2968.0	4152.0	8190.4
10 -- 36	10.0	177.7	267.3	519.7	1777.0	2673.0	5197.0
11 -- 12	16.0	209.9	235.1	487.5	3358.4	3761.6	7800.0
12 -- 37	64.0	298.8	146.2	398.6	19123.2	9356.8	25510.4
13 -- 39	28.0	126.7	318.5	570.7	3547.6	8918.0	15979.6
13 -- 53	54.0	173.9	271.1	523.5	9390.6	14639.4	28269.0
14 -- 40	10.0	133.3	283.4	133.3	1333.0	2834.0	1333.0
14 -- 41	46.0	154.5	304.6	154.5	7107.0	14011.6	7107.0
15 -- 16	140.0	144.7	294.8	144.7	20258.0	41272.0	20258.0
15 -- 42	58.0	128.7	278.8	128.7	7464.6	16170.4	7464.6
16 -- 17	120.0	169.7	319.8	169.7	20364.0	38376.0	20364.0
17 -- 43	48.0	194.7	344.8	194.7	9345.6	16550.4	9345.6
18 -- 54	148.0	247.7	144.2	141.6	36659.6	21341.6	20956.8
19 -- 44	134.0	273.5	170.0	167.4	36649.0	22780.0	22431.6
20 -- 24	198.1	279.2	464.4	0.0	55309.5	91997.6	0.0
20 -- 46	24.0	180.2	464.4	310.8	4324.8	11145.6	7459.2
20 -- 50	198.1	0.0	0.0	211.8	0.0	0.0	41957.6
21 -- 48	134.0	106.6	0.0	491.5	14284.4	0.0	65861.0
21 -- 52	114.0	0.0	156.3	0.0	0.0	17818.2	0.0
22 -- 47	134.0	0.0	182.1	0.0	0.0	24401.4	0.0
22 -- 51	195.0	80.8	0.0	465.7	15756.0	0.0	90811.5
23 -- 50	234.0	0.0	490.2	0.0	0.0	114706.8	0.0
23 -- 55	198.0	0.0	0.0	156.2	0.0	0.0	30927.6
24 -- 49	234.0	0.0	0.0	186.0	0.0	0.0	43524.0
25 -- 57	16.0	450.0	305.0	313.7	7200.0	4880.0	5019.2
25 -- 58	50.0	413.5	268.5	350.2	20675.0	13425.0	17510.0

26 -- 58	51.5	35.3	185.2	31.0	1817.9	9537.8
27 -- 56	50.0	209.7	35.6	18.8	10485.0	1780.0
28 -- 29	10.0	234.1	138.7	60.7	2341.0	1387.0
29 -- 55	39.0	156.7	61.3	138.1	6111.3	2390.7
30 -- 31	11.8	508.2	598.0	277.6	5996.8	7056.4
32 -- 60	179.8	5.8	5.8	2.1	1042.8	1042.8
33 -- 34	2.0	370.1	75.2	319.4	740.2	150.4
34 -- 61	74.0	366.9	72.0	322.6	27150.6	5328.0
35 -- 61	38.9	369.0	74.1	328.4	14354.1	2882.5
35 -- 62	33.8	335.5	40.6	361.9	11339.9	1372.3
36 -- 62	18.0	184.1	260.9	513.3	3313.8	4696.2
37 -- 38	132.0	285.3	159.7	412.1	37659.6	21080.4
38 -- 39	178.0	206.0	239.0	491.4	36668.0	42542.0
40 -- 62	117.5	151.4	301.5	151.4	17789.5	35426.3
41 -- 42	78.0	142.2	292.3	142.2	11091.6	22799.4
43 -- 52	63.3	6.2	156.3	6.2	392.5	9893.8
45 -- 51	58.0	299.3	195.8	193.2	17359.4	11356.4
46 -- 51	122.0	218.5	0.0	272.5	26657.0	0.0
47 -- 52	114.0	132.4	0.0	517.3	15093.6	0.0
48 -- 51	195.0	0.0	207.9	0.0	0.0	40540.5
49 -- 55	198.0	334.8	520.0	0.0	66290.4	102960.0
50 -- 23	234.0	305.0	0.0	0.0	71370.0	0.0
52 -- 53	38.3	138.6	0.0	523.5	5308.4	0.0
53 -- 54	64.7	312.5	271.1	0.0	20218.8	17540.2
56 -- 59	66.0	22.8	172.7	43.5	1504.8	11398.2
58 -- 60	52.0	378.2	83.3	319.2	19666.4	4331.6

TOTAL LOADFEET

988865.85 992624.5



TABLE 11. CASE 2 - MAXIMUM QUEUE SIZE ALGORITHM (BASIC LAYOUT)

# AGVs	AVERAGE LOADED TRAVEL TIME (%/VEHICLE)	AVERAGE UNLOADED TRAVEL TIME (%/VEHICLE)	AVERAGE LOADING/ UNLOADING TIME (%/VEHICLE)	AVERAGE PARKING TIME (%/VEHICLE)	AVERAGE WAITING TIME (%/VEHICLE)	AVERAGE SYSTEM THROUGHPUT (UNITS)
18	43.2	33.7	23.1	0.0	17.8	800
19	42.9	34.5	22.6	0.0	19.6	823
20	43.0	34.6	22.5	0.0	19.8	863
21	43.3	34.5	22.3	0.0	21.4	899
22	43.3	35.5	21.5	0.0	23.4	912
23	43.1	35.5	21.4	0.0	24.2	941
24	43.3	35.4	21.3	0.0	24.3	979

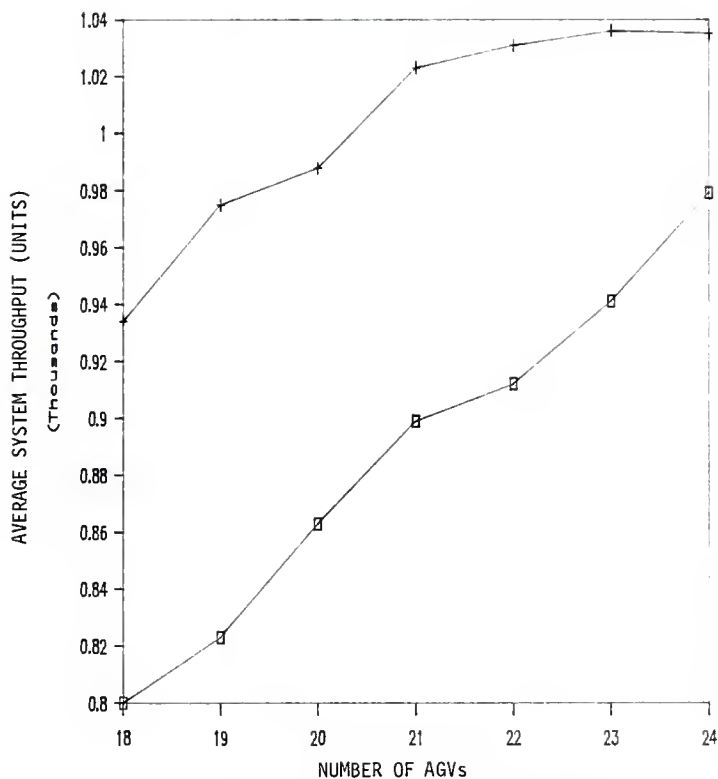
TABLE 12. CASE 2 - SHORTEST DISTANCE ALGORITHM (BASIC LAYOUT)

# AGVs	AVERAGE LOADED TRAVEL TIME (%/VEHICLE)	AVERAGE UNLOADED TRAVEL TIME (%/VEHICLE)	AVERAGE LOADING/ UNLOADING TIME (%/VEHICLE)	AVERAGE PARKING TIME (%/VEHICLE)	AVERAGE WAITING TIME (%/VEHICLE)	AVERAGE SYSTEM THROUGHPUT (UNITS)
18	57.1	15.9	27.0	0.0	20.9	934
19	55.1	18.3	26.7	0.0	20.1	975
20	55.5	18.7	25.8	0.0	22.6	988
21	53.5	21.1	25.4	0.0	26.6	1023
22	50.6	25.0	24.3	0.1	23.5	1031
23	48.5	27.9	23.5	0.1	24.2	1036
24	47.6	29.8	22.5	0.1	24.5	1035

TABLE 13. CASE 1 - PARTS WAITING TIME (BASIC LAYOUT)

# AGVs	AVERAGE WAITING TIME (minute)		MINIMUM WAITING TIME (minute)		MAXIMUM WAITING TIME (minute)	
	MSQA	SDA	MSQA	SDA	MSQA	SDA
18	52.3	21.0	4.7	1.2	372.3	257.8
19	47.3	14.3	4.7	1.2	291.2	228.7
20	41.6	13.1	4.7	1.2	332.8	188.0
21	35.1	10.5	4.7	1.1	281.7	265.5
22	34.9	8.5	4.7	1.1	284.2	130.0
23	29.4	7.2	4.7	1.0	231.9	123.9
24	21.4	6.3	4.7	1.1	108.0	64.0

( SDA = Shortest Distance Algorithm  
 MSQA = Maximum Queue Size Algorithm )



LEGEND: + SHORTEST DISTANCE

MAXIMUM QUEUE SIZE

FIGURE 10. CASE 2 - THROUGHPUT (BASIC LAYOUT)

algorithm as found in CASE 1. We also noted that the rate of change of the throughput for the SDA between 21 and 24 AGVs diminished as compared to that between 18 to 20 AGVs.

From the observation of the animation, shortcuts were introduced to the system as shown in Figure 11. The total loadfeet for the improved guidepath was 718992.9 and the range number of AGVs was 15 to 20. Nineteen AGVs were needed to satisfy the requirements when SDA was used, and more than 20 AGVs were needed when MQSA was used.

The average throughput, and the average percent loaded, unloaded travel time, loading/unloading time, parking and waiting time (per vehicle) were shown in Table 14 and 15. Table 16 tabulated the average, minimum and maximum parts waiting time. The average throughput for both algorithm was plotted in Figure 12. Again, the SDA delivered more parts than the MQSA, and the rate of change of the throughput for the SDA between 18 AGVs and 20 AGVs diminished as compared to that between 15 and 18 AGVs because there were fewer parts to be transported.

From the results, one could notice that as the layout became more complicated, the locations of the stations were more widespread, and as the location of input station with most jobs were widely spread, the shortest distance algorithm will perform better than the maximum queue size algorithm.

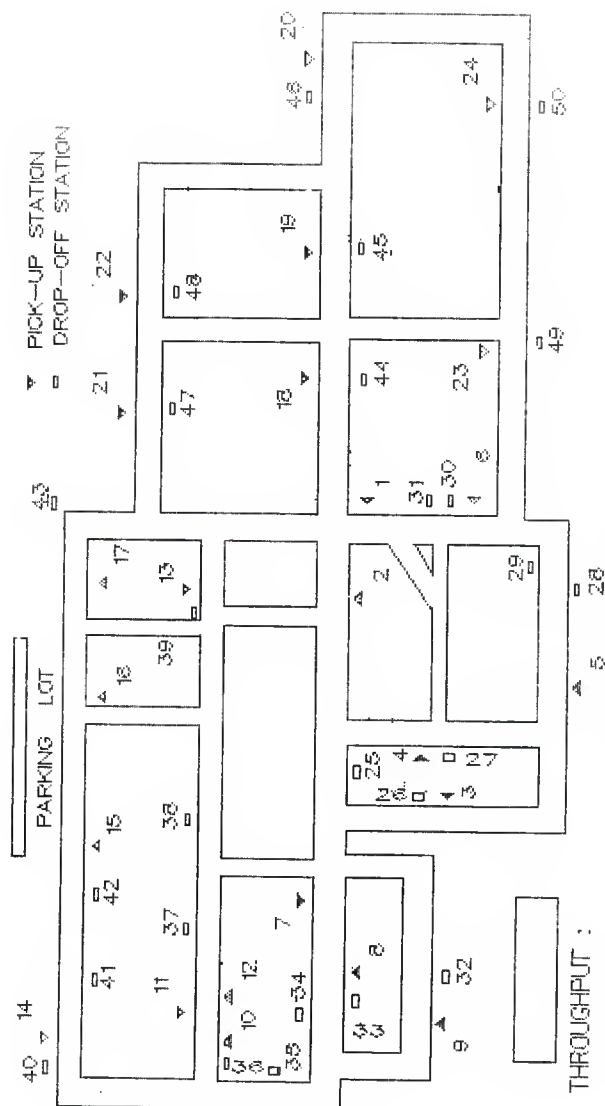


FIGURE 11. CASE 2 - IMPROVED LAYOUT

TABLE 14. CASE 2 - MAXIMUM QUEUE SIZE ALGORITHM (IMPROVED LAYOUT)

# AGVs	AVERAGE LOADED TRAVEL TIME (%/VEHICLE)	AVERAGE UNLOADED TRAVEL TIME (%/VEHICLE)	AVERAGE LOADING/ UNLOADING TIME (%/VEHICLE)	AVERAGE PARKING TIME (%/VEHICLE)	AVERAGE WAITING TIME (%/VEHICLE)	AVERAGE SYSTEM THROUGHPUT (UNITS)
15	39.4	33.1	27.4	0.0	16.8	792
16	39.3	33.7	27.0	0.0	17.9	831
17	40.3	33.0	26.9	0.0	19.6	873
18	39.5	34.2	26.3	0.0	20.5	909
19	39.2	34.6	26.1	0.0	21.8	954
20	39.4	34.7	25.8	0.0	22.3	993

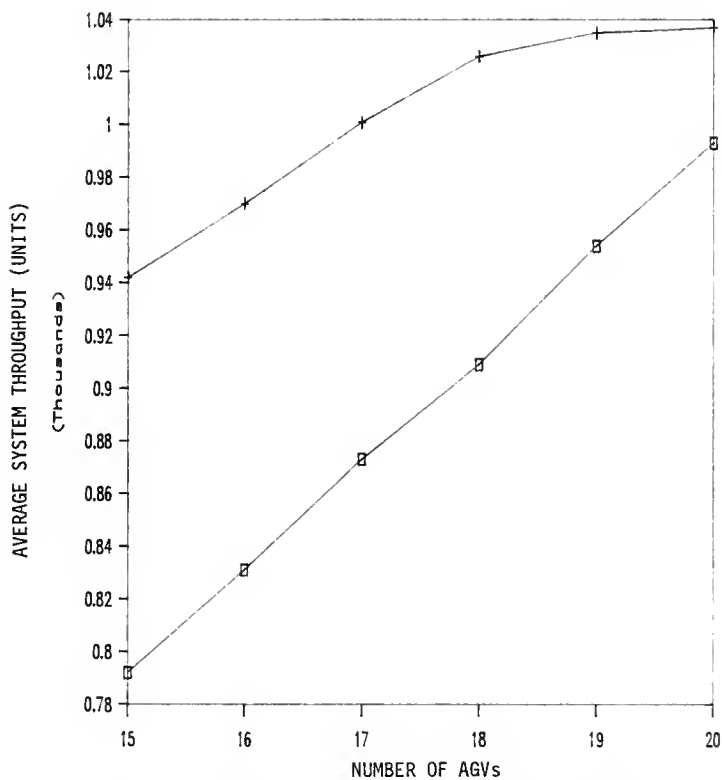
TABLE 15. CASE 2 - SHORTEST DISTANCE ALGORITHM (IMPROVED LAYOUT)

# AGVs	AVERAGE LOADED TRAVEL TIME (%/VEHICLE)	AVERAGE UNLOADED TRAVEL TIME (%/VEHICLE)	AVERAGE LOADING/ UNLOADING TIME (%/VEHICLE)	AVERAGE PARKING TIME (%/VEHICLE)	AVERAGE WAITING TIME (%/VEHICLE)	AVERAGE SYSTEM THROUGHPUT (UNITS)
15	50.2	17.2	32.6	0.0	19.5	942
16	49.1	19.4	31.5	0.0	20.1	970
17	48.5	20.8	30.7	0.0	20.8	1001
18	47.4	22.9	29.7	0.0	22.3	1026
19	45.7	26.0	28.3	0.0	22.5	1035
20	42.8	30.3	27.0	0.0	22.7	1037

TABLE 16. CASE 1 - PARTS WAITING TIME (IMPROVED LAYOUT)

# AGVs	AVERAGE WAITING TIME (minute)		MINIMUM WAITING TIME (minute)		MAXIMUM WAITING TIME (minute)	
	MSQA	SDA	MSQA	SDA	MSQA	SDA
15	52.4	21.8	3.4	1.1	373.7	242.7
16	45.4	18.1	3.6	1.1	272.6	227.1
17	39.7	13.8	3.6	1.2	339.1	255.3
18	32.6	9.3	2.9	1.2	251.5	95.4
19	25.6	6.1	2.4	1.2	101.1	59.1
20	17.5	5.5	2.5	1.1	193.2	43.0

( SDA = Shortest Distance Algorithm  
 MSQA = Maximum Queue Size Algorithm )



LEGEND: + SHORTEST DISTANCE

MAXIMUM QUEUE SIZE

FIGURE 12. CASE 2 - THROUGHPUT (IMPROVED LAYOUT)



## CONCLUSION

The animation developed for the AGVS has proven to be an effective tool for investigating traffic congestions. By observing the behavior of the system from the animation, we can make effective changes, such as shortcuts and possible relocation of stations, with little efforts. Animation can also be used to see the effect of a change to the system (such as the effects of different selection algorithms), and to verify our model or programming logic. It is also very helpful in debugging programs, too.

The only point that one may question is the effort required to build the animation. The CINEMA animation can be built readily when the model is relatively small or medium sized. As the system size increases, the animation construction becomes more difficult.

Even though most of the difficulties can be attributed to the size of the screen, the size and orientation of CINEMA symbols representing the stations and queues have made it difficult to refine CINEMA resolution. Figure 13 shows the actual size for the station and queue symbol.

The station symbol is oriented horizontally, and the problem arises when there are several horizontally adjacent stations. Due to the size of the station symbols, the symbols will overlap each other. When defining a route or a segment between two stations, CINEMA will acknowledge the

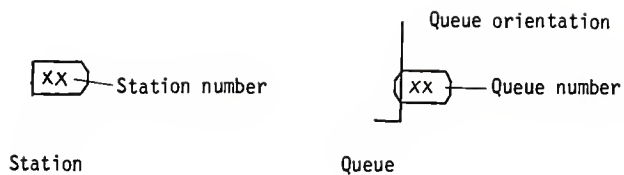
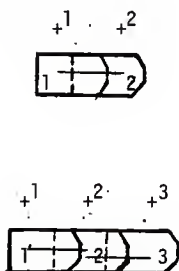
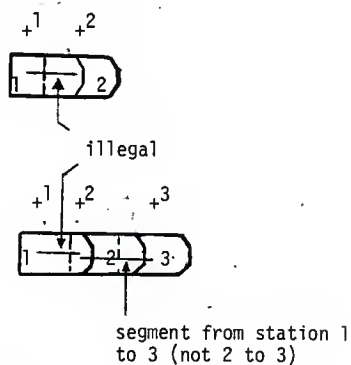


FIGURE 13. CINEMA SYMBOLS FOR STATION AND QUEUE

Good definition



Bad definition



( + = physical location of the stations )

FIGURE 14. PROBLEMS WITH STATION SYMBOLS IN DEFINING THE ROUTE

overlapping station only at the location when it does not overlap the first one (for example, see Figure 14).

This problem can be overcome if we can spread the location of adjacent stations. However, this will result in a crude animation. In the animation of the AGVS, we observed that the AGV traveled faster in some segments and slower in other segments as a result of spreading the location of the adjacent stations apart. This also makes it impossible to decrease the length of the segments though they can still be reduced and be valid in the simulation model. Another disadvantage is that a station can be used (defined) in more than one place.

The queues can be placed in different orientations. However, these orientations do not affect the orientation of the entities (parts) staying in the queues (see Figure 15). The other problem with the queue is the orientation of the queue number. A queue can be placed in different orientation, but its queue number is always horizontally oriented. Although this does not cause as severe a problem as the station symbols, it complicates the construction of the layout.

The limit on the maximum number of queues (max. 200) that can be used in the animation is another disadvantage of CINEMA. The problem of exceeding the number of queues can partly be attributed to SIMAN. A number of SIMAN commands require that a queue be placed before them.

Queue orientation

Actual display of  
parts waiting in the  
queue

Preferred display

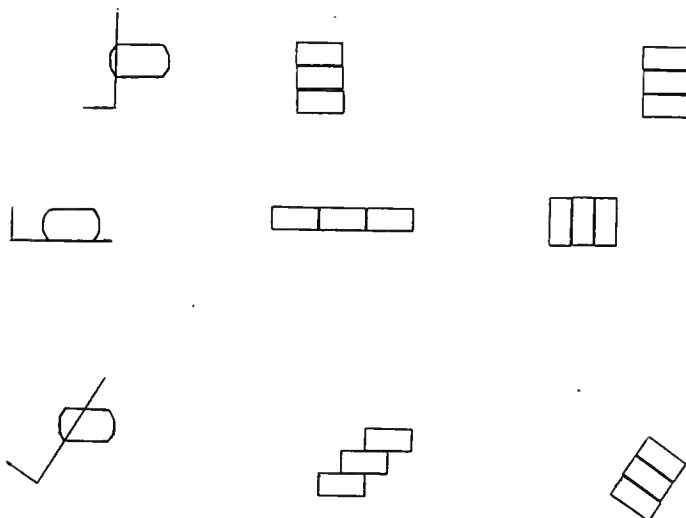


FIGURE 15. ACTUAL vs PREFERRED DISPLAY OF PARTS IN QUEUES

The advantage of the SIMAN framework, which allows us to change the parameter values in the experimental file without changing the model file, also comes with the disadvantage of recompiling the experimental file and relinking both files. The framework would be more effective if any changes in the experimental file can be done interactively.

The problem of relinking both files gives SIMAN discrete event orientation a better edge. Because the size of the model file of discrete event orientation is smaller than the size of the discrete orientation (part of the program is written in the FORTRAN subroutines), the linking time of the discrete event file is much shorter.

From the analysis of both selection algorithms, the shortest distance algorithm would transport more parts than the maximum queue size. As the layout or guidepath became more complicate, the shortest distance algorithm would required fewer number of AGVs than the maximum queue size algorithm. The average, minimum and maximum parts waiting time for the shortest distance algorithm were also smaller than those of the maximum queue size algorithm.

In conclusion, animation is very useful not only to investigate congestion but to debug and verify the model as well. CINEMA/EGA animation can be constructed readily when the model is of small to moderate size. The animation construction gets more complicated as the size of the model

size increases.

## BIBLIOGRAPHY

1. Bischoff, R.A, "The Development of Automated Guided Vehicle Systems for Vertical Stacking/Retrieval Functions", Proceedings of the First International Conference on Automated Guided Vehicles, IFS(Publications) Ltd., 1981, p.53.
2. Egbelu, P.J. and J.M Tanchoco, "Characterization of Automatic Guided Vehicle Dispatching Rules", International Journal of Production Research, vol. 22, no. 3, 1984, pp. 350-374.
3. Hitchen, Max W., "Simulation: The Key to Automation Without Risk", CAD/CAM Technology, Fall 1984, pp. 15-17.
4. Kulwiec, Ray, "Trends in Automated Guided Vehicle Systems", Plant Engineering, October, 1984, pp. 66-73.
5. Morris, E.W., "Developments in Guided Vehicle Systems - Possibilities and Limitations and The Economics of Their Operation", Proceedings of the First International Conference on AGV Systems, IFS(Publications) Ltd., 1981, pp. 67-73.
6. Muller, Ing Thomas, Automated Guided Vehicles, IFS (Publications) Ltd., United Kingdom, 1983.
7. Norman, Susan K., "Design of a Simulation Package for Automated Guided Vehicle Systems", Master's Report, Ohio University, 1985.

8. Pegden, C. Dennis, Introduction to SIMAN, System Modeling Corporation, 1986.
9. Pence, James A and David Finkel, "Modeling a computer system with SIMAN on the IBM Personal Computer", Modeling & Simulation on Microcomputer: Proceeding of the Conference, San Diego, 1986, pp.107-111.
10. Read, Anthony Shoemaker, "A Study of Implementation and Evaluation Techniques of Advance Guided Vehicle Systems", Master's Report, Kansas State University, 1985.
11. Russell, Roberta and Jose Tanchoco, "An evaluation of Vehicle Dispatching Rules and Their Effect on Shop Performance", Material Flow, vol. 1, no. 4, May 1984, pp. 271-280.
12. Rygh, O.B., "New AGVS applications", Proceeding of the Third International Conference on AGV Systems, IFS(Publications) Ltd., 1985, pp. 357-361.
13. Stauffer, Robert N., "Graphic Simulation Answers Preproduction Questions", CAD/CAM Technology, Fall 1984, pp.11-13.
14. System Modeling Corp., "CINEMA/EGA User's Manual", 1986.
15. Vester, John, "The Increasing Importance of AGVs on Inventory Control", P & IM Review, November, 1987, pp. 36-39.



APPENDIX A  
PROGRAM FOR CASE 1

# OHM.MOD & OHMREV.MOD

(CASE 1 BASIC & IMPROVED LAYOUT MODEL FILES)

Both OHM.MOD and OHMREV.MOD use 12 attributes: A(1) - A(12), and 46 variables: X(1) - X(46).

A(1) = Next segment to travel.  
A(2) = length of the segment to be traveled.  
A(3) = Current station number or segment just traveled.  
A(4) = Station of origin, also used as current location of AGV.  
A(5) = Destination of AGV or of part.  
A(6) = AGV status : 0 = picking up part  
                  1 = transporting part to destination  
                  3 = empty and going to parking lot.  
A(7) = not used.  
A(8) = AGV speed.  
A(9) = origin of the AGV : 0 = from parking lot  
                          1 = not from parking lot.  
A(10) = time of part creation.  
A(11) = cinema attribute.  
A(12) = AGV number  
  
X(1) - X(11) = loaded time.  
X(12) - X(22) = unloaded time.  
X(23) - X(33) = waiting time.  
X(34) - X(44) = loading time.  
X(45), X(46) = counters.

```

BEGIN, 1, 1, YES, ohm, NO;
;
;                               OHM.MOD
;                               (CASE 1 - BASIC LAYOUT)
;
; GENERATION OF JOBS FOR EACH INPUT STATION. FROM THE
; GIVEN DATA, THE NUMBER OF JOBS AT EACH STATION IS KNOWN
; AND THE INTER-ARRIVAL TIME BETWEEN JOBS IS TAKEN TO BE
; UNIFORMLY DISTRIBUTED. THE FIRST JOB (AT EACH STATION)
; IS CREATED AT .5% OF ITS EXPECTED INTER-ARRIVAL TIME.
; THE JOBS IS GENERATED WITH RANDOM NUMBER GENERATOR
; STREAM # 2.
;
;
; CREATE, 1, 48:UN(21, 2);
; ASSIGN:A(4)=1;
; BRANCH, 1:ALWAYS, DEC;
; CREATE, 1, 14:UN(22, 2);
; ASSIGN:A(4)=2;
; BRANCH, 1:ALWAYS, DEC;
; CREATE, 1, 7:UN(23, 2);
; ASSIGN:A(4)=3;
; BRANCH, 1:ALWAYS, DEC;
; CREATE, 1, 4:UN(24, 2);
; ASSIGN:A(4)=4;
; BRANCH, 1:ALWAYS, DEC;
; CREATE, 1, 13:UN(25, 2);
; ASSIGN:A(4)=5;
; BRANCH, 1:ALWAYS, DEC;
; CREATE, 1, 15:UN(26, 2);
; ASSIGN:A(4)=6;
; BRANCH, 1:ALWAYS, DEC;
; CREATE, 1, 13:UN(27, 2);
; ASSIGN:A(4)=7;
; BRANCH, 1:ALWAYS, DEC;
; CREATE, 1, 3:UN(28, 2);
; ASSIGN:A(4)=8;
; BRANCH, 1:ALWAYS, DEC;
; CREATE, 1, 5:UN(29, 2);
; ASSIGN:A(4)=9;
; BRANCH, 1:ALWAYS, DEC;
; CREATE, 1, 13:UN(30, 2);
; ASSIGN:A(4)=10;
; BRANCH, 1:ALWAYS, DEC;
; CREATE, 1, 7:UN(31, 2);
; ASSIGN:A(4)=11;
; BRANCH, 1:ALWAYS, DEC;
; CREATE, 1, 11:UN(32, 2);
; ASSIGN:A(4)=12;
; BRANCH, 1:ALWAYS, DEC;
; CREATE, 1, 10:UN(33, 2);
; ASSIGN:A(4)=13;
; BRANCH, 1:ALWAYS, DEC;

```

```

        CREATE,1,3:UN(34,2);
        ASSIGN:A(4)=14;
        BRANCH,1:ALWAYS,DEC;
        CREATE,1,51:UN(35,2);
        ASSIGN:A(4)=15;
        BRANCH,1:ALWAYS,DEC;
        CREATE,1,1:UN(36,2);
        ASSIGN:A(4)=16;
        BRANCH,1:ALWAYS,DEC;
        CREATE,1,8:UN(37,2);
        ASSIGN:A(4)=17;
        BRANCH,1:ALWAYS,DEC;
        CREATE,1,144:UN(38,2);
        ASSIGN:A(4)=18;
;
;   LIMIT THE TOTAL CONCURRENT JOBS IN THE SYSTEM TO 325.
;
DEC      ASSIGN:
        X(45)=X(45)+1;
        BRANCH,1:
        IF,X(45).GT.325,END:
        ELSE,REQ;
;
;   IF TOTAL CONCURRENT JOBS IS > 325, DISPOSE THE INCOMING
;   JOBS.
;
END      ASSIGN:
        X(45)=X(45)-1:DISPOSE;
;
;   REQUEST FOR AN AGV (SEND A SIGNAL).  IF THERE IS AN AGV
;   IN PARKING LOT, REQUEST THAT AGV.  IF NOT, WAIT FOR NEXT
;   AVAILABLE AGV.
;
REQ      SIGNAL:1;
        ASSIGN:M=A(4);
        ASSIGN:A(11)=26;
;
;   WAIT UNTIL AN AGV IS ALLOCATED.
;
        QUEUE,M;
        WAIT:M+25,1;
        ASSIGN:A(11)=27;
;
;   IF AN AGV HAS BEEN ALLOCATED, WAIT UNTIL IT REACHES THE
;   JOB.
;
        QUEUE,M+120;
        WAIT:M+105,1;
;
;   GATHER STATISTIC FOR THE TIME BETWEEN JOB CREATION
;   AND THE TIME THE JOB IS PICKED UP, AND DECREMENT THE

```

```

; TOTAL CONCURRENT JOBS IN THE SYSTEM.
;
; TALLY:1,INT(10);
; ASSIGN:
; X(45)=X(45)-1:DISPOSE;
;
; GENERATION OF AGVs. THE AGVs IS CREATED AT THE
; BEGINNING OF THE SIMULATION.
;
; CREATE,1:0,7;
; ASSIGN:A(11)=29;
; ASSIGN:X(46)=X(46)+1;
; ASSIGN:A(12)=X(46);
;
; WAIT FOR A REQUEST.
;
; WAIT ASSIGN:A(11)=29;
; QUEUE,140;
; WAIT:1,1;
;
; IF THE SIGNAL FOR A REQUEST (JOB) IS ACCEPTED, FIND THE
; JOB USING THE MAXIMUM QUEUE SIZE SELECTION RULE. IF
; THE JOB IS FOUND, GET THE JOB. IF NOT, WAIT FOR ANOTHER
; REQUEST.
; NOTE: THIS BLOCK WILL BE SUBSTITUTED WITH A FORTRAN
; SUBROUTINE (USING AN EVENT BLOCK) WHEN THE
; SHORTEST DISTANCE ALGORITHM IS USED.
;
; FIND FINDJ,1,18:
; MAX(NQ(J));
; BRANCH,1:
; IF,NQ(J).EQ.0,WAIT:
; ELSE,OUT;
;
; IF THE JOB IS FOUND, SEND A SIGNAL TO THE JOB TO
; INDICATE THAT AN AGV HAS BEEN ALLOCATED.
;
; OUT ASSIGN:A(5)=J;
; SIGNAL:A(5)+25;
;
; INITIALIZE THE ROUTE, ASSIGN THE SPEED AND THE CINEMA
; SYMBOL FOR EMPTY AGV, AND SET ON EMPTY TRAVEL TIME.
;
; ASSIGN:A(9)=0;
; ASSIGN:A(11)=28;
; ASSIGN:J=A(12)+11;
; ASSIGN:X(J)=1;
; ASSIGN:M=102;
; ASSIGN:A(4)=A(12)+22;
; ASSIGN:X(A(4))=1;
;

```

```

;      IF THERE IS ANOTHER AGV GOING OUT OF PARKING LOT, WAIT
;      UNTIL THAT AGV GETS OUT OF THE PARKING LOT.
;
      QUEUE,141;
      SEIZE:PATH(102);
      ASSIGN:X(A(4))=0;
      DELAY:UN(20,1),M;
      ASSIGN:NS=35;
      ASSIGN:A(1)=9;
      ASSIGN:IS=0;
      ASSIGN:A(8)=5.0;
      ASSIGN:A(6)=0;
      ROUTE:5,35;
;
;      STATION (MACRO) --
;      CHECK (1) IF THE AGV JUST GOT OUT OF PARKING LOT
;      (2) IF AN INTERSECTION IS REACHED. IF IT'S,
;      ASSIGN THE ROUTE TO DESTINATION.
;
      STATION,1-100;
      BRANCH,1:
        IF,(M.EQ.35).AND.(A(9).EQ.0),OPKR:
        IF,(M.GE.36).AND.(M.LE.41),NRT:
        ELSE,NXT0;
NXT0      ASSIGN:A(3)=M;
NXT2      ASSIGN:M=A(1);
;
;      SET ON WAITING TIME, SEIZE THE NEXT SEGMENT. IF THE NEXT
;      SEGMENT IS OCCUPIED, WAIT UNTIL IT'S FREED. OTHERWISE,
;      SEIZE THE SEGMENT, AND RELEASE THE OCCUPIED SEGMENT.
;
      ASSIGN:A(4)=A(12)+22;
      ASSIGN:X(A(4))=1;
NBLK      QUEUE,M+20;
      SEIZE:PATH(M);
      ASSIGN:X(A(4))=0;
      RELEASE:PATH(A(3));
      ASSIGN:M=A(3);
;
;      CHECK IF (1) THE AGV JUST GOT OUT OF PARKING LOT
;      (2) THE AGV IS EMPTY (WITHOUT A REQUEST)
;      (3) THE DESTINATION IS REACHED
;      IF NOT (1), (2), OR (3), PROCEED TO NEXT STATION.
;
      BRANCH,1:
        IF,A(9).EQ.0,STA:
        IF,A(6).EQ.3,CHK:
        IF,A(5).EQ.A(3),ARR:
        ELSE,GO;
STA      ASSIGN:A(9)=1;
      ASSIGN:M=35;

```

```

GO      ROUTE:A(2)/A(8),SEQ;
;
;      ARRIVAL -- COLLECT STATISTICS FOR EITHER EMPTY OR LOADED
;      TRAVEL, AND PICK UP OR DROP OFF THE JOB.
;
ARR      ASSIGN:X(A(12))=0;
          ASSIGN:J=A(12)+11;
          ASSIGN:X(J)=0;
;
;      GATHER STATISTICS FOR LOADING OR UNLOADING TIME AND
;      ASSIGN ROUTE FROM THE DESTINATION JUST REACHED.
;
          ASSIGN:A(4)=A(12)+33;
          ASSIGN:X(A(4))=1;
          DELAY:UN(19,1),M;
          ASSIGN:X(A(4))=0;
          ASSIGN:NS=A(5);
          ASSIGN:IS=0;
          BRANCH,1:
            IF,A(6).EQ.1,DROP:
            ELSE,DELV;
;
;      DELIVERY -- SET ON LOADED TRAVEL TIME, AND ASSIGN THE
;      DESTINATION OF THE JOB ACCORDING TO THE GIVEN
;      PROBABILITY (FROM THE DATA).
;
DELV      ASSIGN:A(8)=4.0;
          ASSIGN:X(A(12))=1;
          ASSIGN:A(6)=1;
          ASSIGN:A(4)=A(5);
          ASSIGN:J=A(4)+105;
          SIGNAL:J;
          ASSIGN:A(5)=DP(A(4),1);
;
;      ASSIGN CINEMA SYMBOL ACCORDING TO THE STATION ORIGIN AND
;      THE DESTINATION OF THE JOB.
;
          BRANCH,1:ALWAYS,CNM;
;
;      DROP-OFF -- SET ON UNLOADED TRAVEL TIME, AND CHECK IF
;      THERE IS ANYMORE REQUEST.
;
DROP      ASSIGN:A(8)=5.0;
          ASSIGN:J=A(12)+11;
          ASSIGN:X(J)=1;
          ASSIGN:A(6)=0;
          ASSIGN:A(11)=28;
          COUNT:1;
;
;      FIND ANOTHER JOB. IF THERE IS NONE, SEND THE AGV TO THE
;      PARKING LOT AND CHECK FOR MORE JOB ON THE WAY TO PARKING

```

```

;   LOT.  IF PARKING LOT IS REACHED, PARK THE AGV.  IF NOT,
;   SEND THE AGV TO THE NEXT STATION.
;   NOTE:  THE FOLLOWING BLOCKS WILL BE SUBSTITUTED WITH A
;   FORTRAN SUBROUTINE (USING AN EVENT BLOCK) WHEN
;   THE SHORTEST DISTANCE ALGORITHM IS USED TO FIND
;   THE NEXT JOB.
;
CHK      FINDJ,1,18:MAX(NQ(J));
        BRANCH,1:
            IF,(NQ(J).EQ.0).AND.(M.EQ.35),S100:
            IF,NQ(J).EQ.0,PKR:
            ELSE,PORD;
;
;   PARKING STATION -- WAIT IF THERE IS ANOTHER AGV IS,
;   PARKING, AND SET OFF UNLOADED TRAVEL TIME.
;
S100     ASSIGN:A(4)=A(12)+22;
        ASSIGN:X(A(4))=1;
        QUEUE,142;
        SEIZE:PATH(101);
        RELEASE:PATH(9);
        ASSIGN:X(A(4))=0;
        ASSIGN:M=35;
        ROUTE:5,101;
        STATION,101;
        RELEASE:PATH(101);
        DELAY:UN(20,1),M;
        ASSIGN:J=A(12)+11;
        ASSIGN:X(J)=0;
        BRANCH,1:ALWAYS,FIND;
;
;   IF ANOTHER JOB IS FOUND, PICK UP THE JOB.
;
PORD     ASSIGN:A(5)=J;
        SIGNAL:A(5)+25;
        ASSIGN:A(6)=0;
        ASSIGN:A(11)=28;
        BRANCH,1:
            IF,M.EQ.A(5),ARR:
            ELSE,GO;
;
;   IF NO MORE JOB IS FOUND, SEND THE AGV TO PARKING LOT.
;
PKR      ASSIGN:A(5)=35;
        ASSIGN:A(6)=3;
        ASSIGN:A(11)=29;
        ROUTE:A(2)/A(8),SEQ;
OPKR     ASSIGN:A(3)=102;
        BRANCH,1:ALWAYS,NXT2;
;
;   THE FOLLOWING BLOCKS ASSIGN THE SHORTEST ROUTE TO THE

```



```

; DESTINATION (FROM AN INTERSECTION).
; NOTE: THESE BLOCKS WILL BE SUBSTITUTED WITH A FORTRAN
; SUBROUTINE (USING AN EVENT BLOCK) WHEN THE
; SHORTEST DISTANCE ALGORTIHM IS USED.
;
NRT ASSIGN:IS=0;
    BRANCH,1:
        IF,M.EQ.36,C36:
        IF,M.EQ.37,C37:
        IF,M.EQ.38,C38:
        IF,M.EQ.39,C39:
        IF,M.EQ.40,C40:
        IF,M.EQ.41,C41;
C36 ASSIGN:NS=36;
    BRANCH,1:ALWAYS,NXT0;
C37 ASSIGN:NS=37;
    BRANCH,1:ALWAYS,NXT0;
C38 ASSIGN:NS=44;
    ASSIGN:A(1)=43;
    BRANCH,1:
        IF,(A(5).GE.5).AND.(A(5).LE.8),J38A:
        ELSE,NXT0;
J38A ASSIGN:NS=45;
    ASSIGN:A(1)=5;
    BRANCH,1:ALWAYS,NXT0;
C39 ASSIGN:NS=38;
    ASSIGN:A(1)=68;
    BRANCH,1:
        IF,(A(5).EQ.13).OR.(A(5).EQ.30),J39A:
        ELSE,NXT0;
J39A ASSIGN:NS=39;
    ASSIGN:A(1)=69;
    BRANCH,1:ALWAYS,NXT0;
C40 ASSIGN:NS=40;
    ASSIGN:A(1)=63;
    BRANCH,1:
        IF,(A(5).EQ.17).OR.(A(5).EQ.18),J40A:
        IF,(A(5).EQ.33).OR.(A(5).EQ.34),J40A:
        ELSE,NXT0;
J40A ASSIGN:NS=41;
    ASSIGN:A(1)=72;
    BRANCH,1:ALWAYS,NXT0;
C41 ASSIGN:NS=42;
    ASSIGN:A(1)=2;
    BRANCH,1:
        IF,(A(5).EQ.3).OR.(A(5).EQ.21),J41A:
        ELSE,NXT0;
J41A ASSIGN:NS=43;
    ASSIGN:A(1)=70;
    BRANCH,1:ALWAYS,NXT0;
;

```

```

;      THE FOLLOWING BLOCKS ASSIGN THE CINEMA SYMBOL ACCORDING
;      TO THE STATION ORIGIN AND THE DESTINATION OF THE JOB.
;      NOTE: THESE BLOCKS WILL BE SUBSTITUTED WITH A FORTRAN
;            SUBROUTINE (AN EVENT BLOCK) WHEN THE SHORTEST
;            DISTANCE ALGORITHM IS USED.
;
CNM      BRANCH,1:
          IF,(A(4).GE.9).AND.(A(4).LE.11),ONE:
          IF,(A(4).GE.12).AND.(A(4).LE.14),TWO:
          IF,(A(4).GE.15).AND.(A(4).LE.18),THR:
          IF,(A(4).GE.1).AND.(A(4).LE.3),FOUR:
          ELSE,FIVE;
ONE      ASSIGN:J=0;
          BRANCH,1:ALWAYS,DEST;
TWO      ASSIGN:J=1;
          BRANCH,1:ALWAYS,DEST;
THR      ASSIGN:J=2;
          BRANCH,1:ALWAYS,DEST;
FOUR     ASSIGN:J=3;
          BRANCH,1:ALWAYS,DEST;
FIVE     ASSIGN:J=4;
          BRANCH,1:ALWAYS,DEST;
DEST     BRANCH,1:
          IF,(A(5).GE.25).AND.(A(5).LE.28),CIN1:
          IF,(A(5).GE.29).AND.(A(5).LE.31),CIN2:
          IF,(A(5).GE.32).AND.(A(5).LE.34),CIN3:
          IF,(A(5).GE.19).AND.(A(5).LE.21),CIN4:
          ELSE,CIN5;
CIN1     ASSIGN:A(11)=J*5+1;
          ROUTE:A(2)/A(8),SEQ;
CIN2     ASSIGN:A(11)=J*5+2;
          ROUTE:A(2)/A(8),SEQ;
CIN3     ASSIGN:A(11)=J*5+3;
          ROUTE:A(2)/A(8),SEQ;
CIN4     ASSIGN:A(11)=J*5+4;
          ROUTE:A(2)/A(8),SEQ;
CIN5     ASSIGN:A(11)=J*5+5;
          ROUTE:A(2)/A(8),SEQ;
END;

```

```

      SUBROUTINE EVENT(JOB,I)
C*****
C:      SOHM.FOR
C      (CASE 1 - BASIC LAYOUT)
C
C      THIS SUBROUTINE DETERMINES WHICH EVENT OR WHICH ONE OF
C      THE FOLLOWING THREE SUBROUTINES WILL BE EXECUTED
C      THESE SUBROUTINES ARE USED FOR THE SHORTEST DISTANCE
C      ALGORITHM.
C
C*****
      GOTO (1,2,3),I
      1 CALL CINDEF(JOB)
      RETURN
      2 CALL RSDEF(JOB)
      RETURN
      3 CALL SHORSTDST(JOB)
      END
C
C
      SUBROUTINE CINDEF(JOB)
C*****
C      THIS SUBROUTINE ASSIGN THE CINEMA SYMBOL, ACCORDING TO
C      THE STATION ORIGIN AND THE DESTINATION OF THE JOB, TO
C      ATTRIBUTE (11) OF THE JOB.
C
C      ORIG = STATION ORIGIN (STORED IN ATTRIBUTE # 4)
C      DEST = DESTINATION (STORED IN ATTRIBUTE # 5 )
C      CATT = ZONE FOR THE STATION ORIGIN
C      DATT = ZONE FOR THE DESTINATION
C      VAL = THE CINEMA SYMBOL
C
C*****
      COMMON/SIM/D(50),DL(50),S(50),SL(50),X(50),DTNOW,TNOW,
      1TFIN,J,NRUN
      ORIG=A(JOB,4)
      DEST=A(JOB,5)
      IF ((DEST.GE.25).AND.(DEST.LE.28)) THEN
        CATT=1.0
      ELSEIF ((DEST.GE.29).AND.(DEST.LE.31)) THEN
        CATT=2.0
      ELSEIF ((DEST.GE.32).AND.(DEST.LE.34)) THEN
        CATT=3.0
      ELSEIF ((DEST.GE.19).AND.(DEST.LE.21)) THEN
        CATT=4.0
      ELSE
        CATT=5.0
      ENDIF
      IF ((ORIG.GE.9).AND.(ORIG.LE.11)) THEN
        DATT=0.0

```

```

      ELSEIF ((ORIG.GE.12).AND.(ORIG.LE.14)) THEN
          DATT=1.0
      ELSEIF ((ORIG.GE.15).AND.(ORIG.LE.18)) THEN
          DATT=2.0
      ELSEIF ((ORIG.GE.1).AND.(ORIG.LE.3)) THEN
          DATT=3.0
      ELSE
          DATT=4.0
      ENDIF
      VAL=(DATT*5.0)+CATT
      CALL SETA(JOB,11,VAL)
      RETURN
      END
C
C
      SUBROUTINE RSDEF(JOB)
C*****
C
C      THIS SUBROUTINE ASSIGN THE SHORTEST ROUTE THE AGV NEEDS
C      TO TAKE TO GET TO ITS DESTINATION AT EACH INTERSECTION.
C
C      IJUNCT  =  INTERSECTION NUMBER (CURRENT LOCATION OF
C                  THE AGV)
C      DEST    =  FINAL DESTINATION OF THE AGV
C      INUM    =  SEQUENCE (ROUTE) NUMBER
C      VAL     =  NEXT STATION TO BE SEIZE
C
C*****
      COMMON/SIM/D(50),DL(50),S(50),SL(50),X(50),DTNOW,TNOW,
      1TFIN,J,NRUN
      IJUNCT=M(JOB)
      DEST=A(JOB,5)
      IF (IJUNCT.EQ.36) THEN
          INUM=36
          VAL=27
      ELSEIF (IJUNCT.EQ.37) THEN
          INUM=37
          VAL=35
      ELSEIF (IJUNCT.EQ.38) THEN
          IF ((DEST.GE.5).AND.(DEST.LE.8)) THEN
              INUM=45
              VAL=5
          ELSE
              INUM=44
              VAL=43
          ENDIF
      ELSEIF (IJUNCT.EQ.39) THEN
          IF ((DEST.EQ.13).OR.(DEST.EQ.30)) THEN
              INUM=39
              VAL=69
          ELSE

```

```

        INUM=38
        VAL=68
    ENDIF
ELSEIF (IJUNCT.EQ.40) THEN
    IF ((DEST.EQ.17).OR.(DEST.EQ.18)) THEN
        INUM=41
        VAL=72
    ELSEIF ((DEST.EQ.33).OR.(DEST.EQ.34)) THEN
        INUM=41
        VAL=72
    ELSE
        INUM=40
        VAL=63
    ENDIF
ELSEIF (IJUNCT.EQ.41) THEN
    IF ((DEST.EQ.3).OR.(DEST.EQ.21)) THEN
        INUM=43
        VAL=70
    ELSE
        INUM=42
        VAL=2
    ENDIF
ENDIF
CALL SETNS(JOB,INUM)
CALL SETIS(JOB,0)
CALL SETA(JOB,1,VAL)
RETURN
END

C
C
SUBROUTINE SHORTDST(JOB)
C*****
C
C    THIS SUBROUTINE WILL FIND THE CLOSEST REQUEST TO THE
C    AGV. THIS SUBROUTINE READS DATA FROM FILE "SOHM.DAT"
C    WHICH CONTAINS THE SEQUENCE OF INPUT STATIONS.
C
C    Z(K)      =  AN ARRAY THAT STORE THE INPUT STATION NUMBERS
C    TJOB      =  STATUS OF THE AGV:
C                  0 - PICKING UP A JOB
C                  3 - EMPTY (WITHOUT A REQUEST)
C    LASTDST   =  CURRENT LOCATION OF THE AGV
C    DEST      =  FINAL DESTINATION OF THE AGV
C
C*****
COMMON/SIM/D(50),DL(50),S(50),SL(50),X(50),DTNOW,TNOW,
1TFIN,J,NRUN
DIMENSION Z(18)
OPEN(10,FILE='SOHM.DAT')
REWIND 10
LASTDST=A(JOB,4)

```

```

TJOB=0
33 READ(10,*) IONE,ITWO
   IF ((IONE.EQ.LASTDST).AND.(ITWO.EQ.0)) THEN
       READ(10,*) Z(1),Z(2),Z(3),Z(4),Z(5),Z(6),Z(7),Z(8),Z(9),Z(
       READ(10,*) Z(11),Z(12),Z(13),Z(14),Z(15),Z(16),Z(17),Z(18)
   ELSE
       READ(10,*) IONE,ITWO
       READ(10,*) IONE,ITWO
       GOTO 33
   ENDIF
DEST=35

C
C CHECK IF THERE IS ANY REQUEST AT THE CLOSEST INPUT
C STATION. IF THERE IS NONE, CHECK AT THE NEXT CLOSEST
C INPUT STATIONS. IF THERE IS NO MORE REQUEST, SEND THE
C AGV TO THE PARKING LOT (STATION 35).
C

DO 1 K=1,18
   N=Z(K)
   IF ((NQ(N).NE.0).AND.(DEST.EQ.35)) DEST=N
1 CONTINUE
   IF (DEST.EQ.35) TJOB=3
   CALL SETA(JOB,5,DEST)
   CALL SETA(JOB,6,TJOB)
   RETURN
END

```

\*\*\*\*\*

SOHM.DAT

(CASE 1 - BASIC LAYOUT)

THIS FILE CONTAINS THE DATA TO FIND THE CLOSEST  
STATION WITH REQUEST. THE ORGANIZATION OF THE DATA IS AS  
FOLLOWS: LINE 1 : DROP-OFF OR INTERSECTION STATION, ZERO  
LINE 2 : THE FIRST TEN CLOSEST INPUT STATIONS  
LINE 3 : THE SECOND EIGHT CLOSEST INPUT STATIONS

\*\*\*\*\*

19 0  
1 2 3 4 5 6 7 8 9 10  
11 12 13 14 15 16 17 18  
20 0  
2 3 4 5 6 7 8 9 10 11  
12 13 14 15 16 1 17 18  
21 0  
4 5 6 7 8 9 10 11 12 13  
14 15 16 1 17 18 2 3  
22 0  
9 10 11 12 13 14 15 16 1 17  
18 2 3 4 5 6 7 8  
23 0  
9 10 11 12 13 14 15 16 1 17  
18 2 3 4 5 6 7 8  
24 0  
9 10 11 12 13 14 15 16 1 17  
18 2 3 4 5 6 7 8  
25 0  
10 11 12 13 14 15 16 1 17 18  
2 3 4 5 6 7 8 9  
26 0  
10 11 12 13 14 15 16 1 17 18  
2 3 4 5 6 7 8 9  
27 0  
11 12 13 14 15 16 1 17 18 2  
3 4 5 6 7 8 9 10  
28 0  
11 12 13 14 15 16 1 17 18 2  
3 4 5 6 7 8 9 10  
29 0  
14 15 16 1 17 18 2 3 4 5  
6 7 8 9 10 11 12 13  
30 0  
14 15 16 1 17 18 2 3 4 5  
6 7 8 9 10 11 12 13  
31 0  
15 16 1 17 18 2 3 4 5 6  
7 8 9 10 11 12 13 14  
32 0

16 1 17 18 2 3 4 5 6 7  
 8 9 10 11 12 13 14 15  
 33 0  
 17 18 1 2 3 4 5 6 7 8  
 9 10 11 12 13 14 15 16  
 34 0  
 18 2 3 4 5 6 7 8 9 10  
 11 12 13 14 15 16 1 17  
 35 0  
 9 10 11 12 13 14 15 16 1 17  
 18 2 3 4 5 6 7 8  
 38 0  
 5 6 7 8 9 10 11 12 13 14  
 15 16 1 17 18 2 3 4  
 39 0  
 13 14 15 16 1 17 18 2 3 4  
 5 6 7 8 9 10 11 12  
 40 0  
 16 1 17 18 2 3 4 5 6 7  
 8 9 10 11 12 13 14 15  
 41 0  
 2 3 4 5 6 7 8 9 10 11  
 12 13 14 15 16 1 17 18



```

BEGIN;
;
;           OHM.EXP
;           ( CASE 1 - BASIC LAYOUT)
;
;   GIVE THE TITLE OF THE SUMMARY GENERATED AT THE END OF
;   SIMULATION RUN.
;
PROJECT,CASE I Q D,I. OETOMO,11/26/87;
;
;   DEFINE THE MAX. NUMBER OF CONCURRENT ENTITIES,
;   ATTRIBUTES, QUEUES, AND THE ATTRIBUTE NUMBER FOR CINEMA
;   SYMBOL.
;
DISCRETE,350,15,155,105,11;
;
;   INITIALIZE THE SEED VALUE FOR THE RANDOM NUMBER GENERATOR
;   STREAM (TWO SETS OF SEED VALUES: 3000, 5555 & 7777, 1000,
;   ARE USED TO SIMULATE EACH NUMBER OF AGVs), AND DEFINE
;   ALL PARAMETERS VALUES USED IN THE MODEL FILE.
;
SEEDS:1,3000,2,5555;
PARAMETERS:
1,.27,21,.33,23,.47,25,.6,26,.73,28,.97,29,1,.30:
2,.35,19,.45,22,1,.23:3,.81,19,.83,20,.84,22,.85,25,
.99,26,1,.27:
4,.99,19,1,.20:5,.99,19,1.0,29:6,1,.19:7,.99,19,1,.29:
8,.11,19,.12,20,.15,21,.38,22,.55,23,.57,26,.68,29,.7,30,
.97,31,1,.33:
9,.17,19,.24,21,.26,22,.32,23,.4,26,.5,29,.51,30,.56,31,
.96,33,1,.34:
10,.04,19,.06,21,.26,22,.48,24,.54,29,.55,30,.99,33,1,.34:
11,.59,22,.76,23,.82,29,1,.31:
12,.21,19,.25,20,.25,21,.42,22,.65,23,.72,26,.81,29,.83,30,
.99,32,1,.33:
13,.33,19,.36,21,.46,22,.55,23,.56,24,.62,25,.7,26,.94,28,
.96,31,1,.33:
14,.8,19,.81,21,.83,22,.99,23,1,.26:15,1,.19:
16,.05,20,.21,21,.44,22,.64,23,.65,24,.68,25,.73,26,.75,27,
.76,28,.88,29,.9,30,.92,31,.94,32,.995,33,1,.34:
17,.44,19,.46,22,.71,25,.72,26,.94,28,.98,29,.99,30,1,.31:
18,.2,19,.4,25,.6,27,.8,28,1,.29:19,60,120:20,30,60:
21,8640,10560:22,2541,3106:23,1296,1584:24,762,932:
25,2400,2933:26,2700,3300:27,2400,2933:28,566,692:
29,939,1148:30,2356,2880:31,1194,1460:32,1906,2329:
33,1865,2072:34,456,507:35,9257,11314:36,247,302:
37,1464,1790:38,25920,28800;
;
;   DEFINE THE ROUTE FROM EACH INPUT, DROP-OFF STATION,
;   AND INTERSECTIONS.
;   The SEQUENCE block requires that the sequence be defined

```

```

;   consecutively (1, 2,3, ...). This block has the format
;   of: SEQUENCE:NSEQ,NSTA,A1,A2, .../...:repeats ...;
;   where NSEQ      = sequence set number [integer]
;         NSTA      = next station number in the
;                   visitation sequence [integer]
;         A1,A2, ... = attribute assignment value.
;   This block has two components: NS and IS, where NS is
;   the sequence number and IS is the index of the list of
;   station visitations.
;   For example: SEQUENCE:1,20,62,24/62,73,28: assigns
;   station 20 as the first station to be visited in the
;   sequence set number 1. At station 20, the attribute
;   number 1 and 2 of the job are changed to 62 and 24,
;   respectively, and IS is incremented by 1. IS needs to
;   be set to zero again when the job takes different route.
;   Otherwise, the first station to be visited in its new
;   station visitations sequence will be the nth station,
;   where n equals to the IS value of the job's last
;   sequence.
;

```

#### SEQUENCES:

```

1,20,62,24/62,73,28/73,41,29/41,,29:
2,46,53,12/53,54,12/54,58,29/58,74,29/74,75,29/75,4,29/
  4,76,30/76,38,31/38,,31:
3,21,55,10/55,53,44/53,54,12/54,58,29/58,74,29/74,75,29/
  75,4,29/4,76,30/76,38,31/38,,31:
4,76,38,31/38,,31:
5,6,45,30/45,50,6/50,56,32/56,48,40/48,51,20/51,22,12/
  22,77,16/77,23,25/23,78,25/78,24,30/24,37,30/37,35,27:
6,45,50,6/50,56,32/56,48,40/48,51,20/51,22,12/22,77,16/
  77,23,25/23,78,25/78,24,30/24,37,30/37,35,27:
7,8,47,30/47,48,78/48,51,20/51,22,12/22,77,16/77,23,25/
  23,78,25/78,24,30/24,37,30/37,35,27:
8,47,48,78/48,51,20/51,22,12/22,77,16/77,23,25/23,78,25/
  78,24,30/24,37,30/37,35,27:
9,64,79,34/79,25,34/25,80,34/80,26,31/26,10,31/10,81,24/
  81,36,34/36,27,34:
10,81,36,34/36,27,34:
11,65,66,38/66,82,38/82,83,38/83,12,39/12,67,39/67,84,25/
  84,39,25/39,,26:
12,67,84,25/84,39,25/39,,26:
13,30,59,56/59,71,22/71,88,12/88,14,20/14,89,20/89,31,20/
  31,42,20/42,85,31/85,15,31/15,32,32/32,86,12/86,87,24/
  87,40,24/40,,24:
14,89,31,20/31,42,20/42,85,31/85,15,31/15,32,32/32,86,12/
  86,87,24/87,40,24/40,,24:
15,32,86,12/86,87,24/87,40,24/40,,24:
16,61,52,12/52,88,20/88,19,20/19,1,20/1,20,12/20,62,24/
  62,73,28/73,41,29/41,,29:
17,34,18,56/18,60,36/60,61,6/61,52,12/52,88,20/88,19,20/
  19,1,20/1,20,12/20,62,24/62,73,28/73,41,29/41,,29:

```

18,60,61,6/61,52,12/52,88,20/88,19,20/19,1,20/1,20,12/  
 20,62,24/62,73,28/73,41,29/41,,29:  
 19,1,20,12/20,62,24/62,73,28/73,41,29/41,,29:  
 20,62,73,28/73,41,29/41,,29:  
 21,55,53,44/53,54,12/54,58,29/58,74,29/  
 74,75,29/75,4,29/4,76,30/76,38,31/38,,31:  
 22,77,23,25/23,78,25/78,24,30/24,37,30/37,35,27:  
 23,78,24,30/24,37,30/37,35,27:  
 24,37,35,27:  
 25,80,26,31/26,10,31/10,81,24/81,36,34/36,27,34:  
 26,10,81,24/81,36,34/36,27,34:  
 27,28,11,4/11,65,32/65,66,38/66,82,38/82,83,38/83,12,39/  
 12,67,39/67,84,25/84,39,25/39,,26:  
 28,11,65,32/65,66,38/66,82,38/82,83,38/83,12,39/12,67,39/  
 67,84,25/84,39,25/39,,26:  
 29,71,88,12/88,14,20/14,89,20/89,31,20/31,42,20/42,85,31/  
 85,15,31/15,32,32/32,86,12/86,87,24/87,40,24/40,,24:  
 30,59,71,24/71,88,12/88,14,20/14,89,20/89,31,20/31,42,20/  
 42,85,31/85,15,31/15,32,32/32,86,12/86,87,24/87,40,24/  
 40,,24:  
 31,42,85,31/85,15,31/15,32,32/32,86,12/86,87,24/87,40,24/  
 40,,24:  
 32,86,87,24/87,40,24/40,,24:  
 33,17,34,22/34,18,56/18,60,36/60,61,6/61,52,12/52,88,20/  
 88,19,20/19,1,20/1,20,12/20,62,24/62,73,28/73,41,29/41,,29:  
 34,18,60,36/60,61,6/61,52,12/52,88,20/88,19,20/19,1,20/  
 1,20,12/20,62,24/62,73,28/73,41,29/41,,29:  
 35,9,64,27/64,79,34/79,25,34/25,80,34/80,26,31/26,10,31/  
 10,81,24/81,36,34/36,27,34:  
 36,27,28,34/28,11,4/11,65,32/65,66,38/66,82,38/82,83,38/  
 83,12,39/12,67,39/67,84,25/84,39,25/39,,26:  
 37,35,9,39/9,64,38/64,79,34/79,25,34/25,80,34/80,26,31/  
 26,10,31/10,81,24/81,36,34/36,27,34:  
 38,68,29,12/29,71,24/71,88,12/88,14,20/14,89,20/89,31,20/  
 31,42,20/42,85,31/85,15,31/15,32,32/32,86,12/86,87,24/  
 87,40,24/40,,24:  
 39,69,13,12/13,30,38/30,59,56:  
 40,63,16,12/16,61,30/61,52,12/52,90,20/90,19,20/19,1,20/  
 1,20,12/20,62,24/62,73,28/73,41,29/41,,29:  
 41,72,33,32/33,17,70/17,34,22/34,18,56/18,60,36:  
 42,2,46,12/46,53,12/53,54,12/54,58,29/58,74,29/74,75,29/  
 75,4,29/4,76,30/76,38,31/38,,31:  
 43,70,49,34/49,3,34/3,21,34/21,55,10:  
 44,43,44,38/44,57,38/57,51,38/51,22,12/22,77,16/77,23,25/  
 23,78,25/78,24,30/24,37,30/37,35,27:  
 45,5,6,26/6,45,30/45,7,18/7,8,14/8,47,30;

```

;
;   INITIALIZE ALL VARIABLES USED TO GATHER PERCENT LOADED,
;   UNLOADED TRAVEL, LOADING/UNLOADING, PARKING, AND WAITING
;   TIME
;

```

```

INITIALIZE,X(1)=0,X(2)=0,X(3)=0,X(4)=0,X(5)=0,X(6)=0,X(7)=0,
      X(8)=0,X(9)=0,X(10)=0,X(11)=0,X(12)=0,X(13)=0,X(14)=0,
      X(15)=0,X(16)=0,X(17)=0,X(18)=0,X(19)=0,X(20)=0,X(21)=0,
      X(22)=0,X(23)=0,X(24)=0,X(25)=0,X(26)=0,X(27)=0,X(28)=0,
      X(29)=0,X(30)=0,X(31)=0,X(32)=0,X(33)=0,X(34)=0,X(35)=0,
      X(36)=0,X(37)=0,X(38)=0,X(39)=0,X(40)=0,X(41)=0,X(42)=0,
      X(43)=0,X(44)=0,X(45)=0,X(46)=0;
;
;   PRINT OUT THE DATA COLLECTED DURING THE SIMULATION RUN.
;
TALLIES:1,TIME IN SYSTEM;
COUNTERS:1,THROUGHPUT;
DSTAT:1,X(1),AGV 1 LOADED:2,X(2),AGV 2 LOADED:3,X(3),AGV 3 LOADED:
4,X(4),AGV 4 LOADED:5,X(5),AGV 5 LOADED:6,X(6),AGV 6 LOADED:
7,X(7),AGV 7 LOADED:8,X(8),AGV 8 LOADED:9,X(9),AGV 9 LOADED:
10,X(10),AGV 10 LOADED:11,X(11),AGV 11 LOADED:
12,X(12),AGV 1 UNLOADED:13,X(13),AGV 2 UNLOADED:
14,X(14),AGV 3 UNLOADED:15,X(15),AGV 4 UNLOADED:
16,X(16),AGV 5 UNLOADED:17,X(17),AGV 6 UNLOADED:
18,X(18),AGV 7 UNLOADED:19,X(19),AGV 8 UNLOADED:
20,X(20),AGV 9 UNLOADED:21,X(21),AGV 10 UNLOADED:
22,X(22),AGV 11 UNLOADED:23,X(23),AGV 1 WAITING:
24,X(24),AGV 2 WAITING:25,X(25),AGV 3 WAITING:
26,X(26),AGV 4 WAITING:27,X(27),AGV 5 WAITING:
28,X(28),AGV 6 WAITING:29,X(29),AGV 7 WAITING:
30,X(30),AGV 8 WAITING:31,X(31),AGV 9 WAITING:
32,X(32),AGV 10 WAITING:33,X(33),AGV 11 WAITING:
34,X(34),AGV 1 LOADING:35,X(35),AGV 2 LOADING:
36,X(36),AGV 3 LOADING:37,X(37),AGV 4 LOADING:
38,X(38),AGV 5 LOADING:39,X(39),AGV 6 LOADING:
40,X(40),AGV 7 LOADING:41,X(41),AGV 7 LOADING:
42,X(42),AGV 9 LOADING:43,X(43),AGV 10 LOADING:
44,X(44),AGV 11 LOADING;
;
;   SPECIFY THE NUMBER OF SEGMENTS IN THE SYSTEM AND THE
;   NUMBER AND LENGTH OF THE SIMULATION RUN(S).
;
RESOURCES:1-105,PATH;
REPLICA,1,0,28800;
END;

```

```

BEGIN,1,1,YES,ohmrev,NO;
;
;                                OHMREV.MOD
;                                (CASE 1 - IMPROVED LAYOUT)
;
; GENERATION OF JOBS FOR EACH INPUT STATION. FROM THE
; GIVEN DATA, THE NUMBER OF JOBS AT EACH STATION IS KNOWN
; AND THE INTER-ARRIVAL TIME BETWEEN JOBS IS TAKEN TO BE
; UNIFORMLY DISTRIBUTED. THE FIRST JOB (AT EACH STATION)
; IS CREATED AT .5% OF ITS EXPECTED INTER-ARRIVAL TIME.
; THE JOBS IS GENERATED WITH RANDOM NUMBER GENERATOR
; STREAM # 2.
;
;
; CREATE,1,48:UN(21,2);
; ASSIGN:A(4)=1;
; BRANCH,1:ALWAYS,DEC;
; CREATE,1,14:UN(22,2);
; ASSIGN:A(4)=2;
; BRANCH,1:ALWAYS,DEC;
; CREATE,1,7:UN(23,2);
; ASSIGN:A(4)=3;
; BRANCH,1:ALWAYS,DEC;
; CREATE,1,4:UN(24,2);
; ASSIGN:A(4)=4;
; BRANCH,1:ALWAYS,DEC;
; CREATE,1,13:UN(25,2);
; ASSIGN:A(4)=5;
; BRANCH,1:ALWAYS,DEC;
; CREATE,1,15:UN(26,2);
; ASSIGN:A(4)=6;
; BRANCH,1:ALWAYS,DEC;
; CREATE,1,13:UN(27,2);
; ASSIGN:A(4)=7;
; BRANCH,1:ALWAYS,DEC;
; CREATE,1,3:UN(28,2);
; ASSIGN:A(4)=8;
; BRANCH,1:ALWAYS,DEC;
; CREATE,1,5:UN(29,2);
; ASSIGN:A(4)=9;
; BRANCH,1:ALWAYS,DEC;
; CREATE,1,13:UN(30,2);
; ASSIGN:A(4)=10;
; BRANCH,1:ALWAYS,DEC;
; CREATE,1,7:UN(31,2);
; ASSIGN:A(4)=11;
; BRANCH,1:ALWAYS,DEC;
; CREATE,1,11:UN(32,2);
; ASSIGN:A(4)=12;
; BRANCH,1:ALWAYS,DEC;
; CREATE,1,10:UN(33,2);
; ASSIGN:A(4)=13;
; BRANCH,1:ALWAYS,DEC;

```

```

        CREATE,1,3:UN(34,2);
        ASSIGN:A(4)=14;
        BRANCH,1:ALWAYS,DEC;
        CREATE,1,51:UN(35,2);
        ASSIGN:A(4)=15;
        BRANCH,1:ALWAYS,DEC;
        CREATE,1,1:UN(36,2);
        ASSIGN:A(4)=16;
        BRANCH,1:ALWAYS,DEC;
        CREATE,1,8:UN(37,2);
        ASSIGN:A(4)=17;
        BRANCH,1:ALWAYS,DEC;
        CREATE,1,144:UN(38,2);
        ASSIGN:A(4)=18;
;
;   LIMIT THE TOTAL CONCURRENT JOBS IN THE SYSTEM TO 325.
;
DEC      ASSIGN:
        X(45)=X(45)+1;
        BRANCH,1:
            IF,X(45).GT.325,END:
            ELSE,REQ;
;
;   IF TOTAL CONCURRENT JOBS IS > 325, DISPOSE THE INCOMING
;   JOBS.
;
END      ASSIGN:
        X(45)=X(45)-1:DISPOSE;
;
;   REQUEST FOR AN AGV (SEND A SIGNAL). IF THERE IS AN AGV
;   IN PARKING LOT, REQUEST THAT AGV. IF NOT, WAIT FOR NEXT
;   AVAILABLE AGV.
;
REQ      SIGNAL:1;
        ASSIGN:M=A(4);
        ASSIGN:A(11)=26;
;
;   WAIT UNTIL AN AGV IS ALLOCATED.
;
        QUEUE,M;
        WAIT:M+25,1;
        ASSIGN:A(11)=27;
;
;   IF AN AGV HAS BEEN ALLOCATED, WAIT UNTIL IT REACHES THE
;   JOB.
;
        QUEUE,M+140;
        WAIT:M+105,1;
;
;   GATHER STATISTIC FOR THE TIME BETWEEN JOB CREATION
;   AND THE TIME THE JOB IS PICKED UP, AND DECREMENT THE

```

```

;      TOTAL CONCURRENT JOBS IN THE SYSTEM.
;
;      TALLY:1,INT(10);
;      ASSIGN:
;          X(45)=X(45)-1:DISPOSE;
;
;      GENERATION OF AGVs.  THE AGVs IS CREATED AT THE
;      BEGINNING OF THE SIMULATION.
;
;      CREATE,1:0,7;
;      ASSIGN:A(11)=29;
;      ASSIGN:X(46)=X(46)+1;
;      ASSIGN:A(12)=X(46);
;
;      WAIT FOR A REQUEST.
;
WAIT    ASSIGN:A(11)=29;
        QUEUE,160;
        WAIT:1,1;
;
;      IF THE SIGNAL FOR A REQUEST (JOB) IS ACCEPTED, FIND THE
;      JOB USING THE MAXIMUM QUEUE SIZE SELECTION RULE.  IF
;      THE JOB IS FOUND, GET THE JOB.  IF NOT, WAIT FOR ANOTHER
;      REQUEST.
;      NOTE:  THIS BLOCK WILL BE SUBSTITUTED WITH A FORTRAN
;      SUBROUTINE (USING AN EVENT BLOCK) WHEN THE
;      SHORTEST DISTANCE ALGORITHM IS USED.
;
FIND    FINDJ,1,18:
        MAX(NQ(J));
        BRANCH,1:
        IF,NQ(J).EQ.0,WAIT:
        ELSE,OUT;
;
;      IF THE JOB IS FOUND, SEND A SIGNAL TO THE JOB TO
;      INDICATE THAT AN AGV HAS BEEN ALLOCATED.
;
OUT     ASSIGN:A(5)=J;
        SIGNAL:A(5)+25;
;
;      INITIALIZE THE ROUTE, ASSIGN THE SPEED AND THE CINEMA
;      SYMBOL FOR EMPTY AGV, AND SET ON EMPTY TRAVEL TIME.
;
        ASSIGN:A(9)=0;
        ASSIGN:A(11)=28;
        ASSIGN:J=A(12)+11;
        ASSIGN:X(J)=1;
        ASSIGN:M=120;
        ASSIGN:A(4)=A(12)+22;
        ASSIGN:X(A(4))=1;
;

```

```

; IF THERE IS ANOTHER AGV GOING OUT OF PARKING LOT, WAIT
; UNTIL THAT AGV GETS OUT OF THE PARKING LOT.
;
    QUEUE,161;
    SEIZE:PATH(120);
    ASSIGN:X(A(4))=0;
    DELAY:UN(20,1),M;
    ASSIGN:NS=35;
    ASSIGN:A(1)=9;
    ASSIGN:IS=0;
    ASSIGN:A(8)=5.0;
    ASSIGN:A(6)=0;
    ROUTE:5,35;
;
; STATION (MACRO) --
; CHECK (1) IF THE AGV JUST GOT OUT OF PARKING LOT
; (2) IF AN INTERSECTION IS REACHED. IF IT'S,
; ASSIGN THE ROUTE TO DESTINATION.
;
    STATION,1-119;
    BRANCH,1:
        IF,(M.EQ.35).AND.(A(9).EQ.0),OPKR:
        IF,(M.GE.38).AND.(M.LE.43),NRT:
        ELSE,NXT0;
    NXT0 ASSIGN:A(3)=M;
    NXT2 ASSIGN:M=A(1);
;
; SET ON WAITING TIME, SEIZE THE NEXT SEGMENT. IF THE NEXT
; SEGMENT IS OCCUPIED, WAIT UNTIL IT'S FREED. OTHERWISE,
; SEIZE THE SEGMENT, AND RELEASE THE OCCUPIED SEGMENT.
;
    ASSIGN:A(4)=A(12)+22;
    ASSIGN:X(A(4))=1;
NBLK QUEUE,M+20;
    SEIZE:PATH(M);
    ASSIGN:X(A(4))=0;
    RELEASE:PATH(A(3));
    ASSIGN:M=A(3);
;
; CHECK IF (1) THE AGV JUST GOT OUT OF PARKING LOT
; (2) THE AGV IS EMPTY (WITHOUT A REQUEST)
; (3) THE DESTINATION IS REACHED
; IF NOT (1), (2), OR (3), PROCEED TO NEXT STATION.
;
    BRANCH,1:
        IF,A(9).EQ.0,STA:
        IF,A(6).EQ.3,CHK:
        IF,A(5).EQ.A(3),ARR:
        ELSE,GO;
    STA ASSIGN:A(9)=1;
    ASSIGN:M=35;

```



```

GO      ROUTE:A(2)/A(8),SEQ;
;
;      ARRIVAL -- COLLECT STATISTICS FOR EITHER EMPTY OR LOADED
;      TRAVEL, AND PICK UP OR DROP OFF THE JOB.
;
ARR      ASSIGN:X(A(12))=0;
          ASSIGN:J=A(12)+11;
          ASSIGN:X(J)=0;
;
;      GATHER STATISTICS FOR LOADING OR UNLOADING TIME AND
;      ASSIGN ROUTE FROM THE DESTINATION JUST REACHED.
;
          ASSIGN:A(4)=A(12)+33;
          ASSIGN:X(A(4))=1;
          DELAY:UN(19,1),M;
          ASSIGN:X(A(4))=0;
          ASSIGN:NS=A(5);
          ASSIGN:IS=0;
          BRANCH,1:
            IF,A(6).EQ.1,DROP:
            ELSE,DELV;
;
;      DELIVERY -- SET ON LOADED TRAVEL TIME, AND ASSIGN THE
;      DESTINATION OF THE JOB ACCORDING TO THE GIVEN
;      PROBABILITY (FROM THE DATA).
;
DELV      ASSIGN:A(8)=4.0;
          ASSIGN:X(A(12))=1;
          ASSIGN:A(6)=1;
          ASSIGN:A(4)=A(5);
          ASSIGN:J=A(4)+105;
          SIGNAL:J;
          ASSIGN:A(5)=DP(A(4),1);
;
;      ASSIGN CINEMA SYMBOL ACCORDING TO THE STATION ORIGIN AND
;      THE DESTINATION OF THE JOB.
;
          BRANCH,1:ALWAYS,CNM;
;
;      DROP-OFF -- SET ON UNLOADED TRAVEL TIME, AND CHECK IF
;      THERE IS ANYMORE REQUEST.
;
DROP      ASSIGN:A(8)=5.0;
          ASSIGN:J=A(12)+11;
          ASSIGN:X(J)=1;
          ASSIGN:A(6)=0;
          ASSIGN:A(11)=28;
          COUNT:1;
;
;      FIND ANOTHER JOB. IF THERE IS NONE, SEND THE AGV TO THE
;      PARKING LOT AND CHECK FOR MORE JOB ON THE WAY TO PARKING

```

```

;   LOT.  IF PARKING LOT IS REACHED, PARK THE AGV.  IF NOT,
;   SEND THE AGV TO THE NEXT STATION.
;   NOTE:  THE FOLLOWING BLOCKS WILL BE SUBSTITUTED WITH A
;   FORTRAN SUBROUTINE (USING AN EVENT BLOCK) WHEN
;   THE SHORTEST DISTANCE ALGORITHM IS USED TO FIND
;   THE NEXT JOB.
;
CHK      FINDJ,1,18:MAX(NQ(J));
        BRANCH,1:
            IF,(NQ(J).EQ.0).AND.(M.EQ.35),S100:
            IF,NQ(J).EQ.0,PKR:
            ELSE,PORD;
;
;   PARKING STATION -- WAIT IF THERE IS ANOTHER AGV IS,
;   PARKING, AND SET OFF UNLOADED TRAVEL TIME.
;
S100     ASSIGN:A(4)=A(12)+22;
        ASSIGN:X(A(4))=1;
        QUEUE,162;
        SEIZE:PATH(121);
        RELEASE:PATH(9);
        ASSIGN:X(A(4))=0;
        ASSIGN:M=35;
        ROUTE:5,121;
        STATION,121;
        RELEASE:PATH(121);
        DELAY:UN(20,1),M;
        ASSIGN:J=A(12)+11;
        ASSIGN:X(J)=0;
        BRANCH,1:ALWAYS,FIND;
;
;   IF ANOTHER JOB IS FOUND, PICK UP THE JOB.
;
PORD     ASSIGN:A(5)=J;
        SIGNAL:A(5)+25;
        ASSIGN:A(6)=0;
        ASSIGN:A(11)=28;
        BRANCH,1:
            IF,M.EQ.A(5),ARR:
            ELSE,GO;
;
;   IF NO MORE JOB IS FOUND, SEND THE AGV TO PARKING LOT.
;
PKR      ASSIGN:A(5)=35;
        ASSIGN:A(6)=3;
        ASSIGN:A(11)=29;
        ROUTE:A(2)/A(8),SEQ;
OPKR     ASSIGN:A(3)=120;
        BRANCH,1:ALWAYS,NXT2;
;
;   THE FOLLOWING BLOCKS ASSIGN THE SHORTEST ROUTE TO THE

```

```

; DESTINATION (FROM AN INTERSECTION).
; NOTE: THESE BLOCKS WILL BE SUBSTITUTED WITH A FORTRAN
; SUBROUTINE (USING AN EVENT BLOCK) WHEN THE
; SHORTEST DISTANCE ALGORITHM IS USED.
;
NRT ASSIGN:IS=0;
    BRANCH,1:
        IF,M.EQ.38,C38:
        IF,M.EQ.39,C39:
        IF,M.EQ.40,C40:
        IF,M.EQ.41,C41:
        IF,M.EQ.42,C42:
        IF,M.EQ.43,C43;
C38 ASSIGN:NS=38;
    ASSIGN:A(1)=27;
    BRANCH,1:ALWAYS,NXT0;
C39 ASSIGN:NS=39;
    ASSIGN:A(1)=68;
    BRANCH,1:
        IF,(A(5).EQ.13).OR.(A(5).EQ.30),J39A:
        ELSE,NXT0;
J39A ASSIGN:NS=40;
    ASSIGN:A(1)=69;
    BRANCH,1:ALWAYS,NXT0;
C40 ASSIGN:NS=41;
    ASSIGN:A(1)=75;
    BRANCH,1:
        IF,(A(5).EQ.17).OR.(A(5).EQ.18),J40A:
        IF,(A(5).EQ.33).OR.(A(5).EQ.34),J40A:
        ELSE,NXT0;
J40A ASSIGN:NS=42;
    ASSIGN:A(1)=74;
    BRANCH,1:ALWAYS,NXT0;
C41 ASSIGN:NS=43;
    ASSIGN:A(1)=2;
    BRANCH,1:
        IF,(A(5).EQ.3).OR.(A(5).EQ.21),J41A:
        ELSE,NXT0;
J41A ASSIGN:NS=44;
    ASSIGN:A(1)=70;
    BRANCH,1:ALWAYS,NXT0;
C42 ASSIGN:NS=45;
    ASSIGN:A(1)=83;
    BRANCH,1:
        IF,(A(5).GE.5).AND.(A(5).LE.8),J42A:
        ELSE,NXT0;
J42A ASSIGN:NS=46;
    ASSIGN:A(1)=5;
    BRANCH,1:ALWAYS,NXT0;
C43 ASSIGN:NS=47;
    ASSIGN:A(1)=99;

```

```

      BRANCH,1:
      IF,(A(5).GE.13).AND.(A(5).LE.18),J43A:
      IF,(A(5).GE.29).AND.(A(5).LE.34),J43A:
      IF,(A(5).GE.1).AND.(A(5).LE.3),J43B:
      IF,(A(5).GE.19).AND.(A(5).LE.21),J43B:
      IF,(A(5).GE.4).AND.(A(5).LE.8),J43C:
      IF,(A(5).GE.22).AND.(A(5).LE.24),J43C:
      ELSE,NXT0;
J43A  ASSIGN:NS=48;
      ASSIGN:A(1)=89;
      BRANCH,1:ALWAYS,NXT0;
J43B  ASSIGN:NS=49;
      ASSIGN:A(1)=89;
      BRANCH,1:ALWAYS,NXT0;
J43C  ASSIGN:NS=37;
      ASSIGN:A(1)=89;
      BRANCH,1:ALWAYS,NXT0;
;
;     THE FOLLOWING BLOCKS ASSIGN THE CINEMA SYMBOL ACCORDING
;     TO THE STATION ORIGIN AND THE DESTINATION OF THE JOB.
;     NOTE:  THESE BLOCKS WILL BE SUBSTITUTED WITH A FORTRAN
;           SUBROUTINE (AN EVENT BLOCK) WHEN THE SHORTEST
;           DISTANCE ALGORITHM IS USED.
;
CNM   BRANCH,1:
      IF,(A(4).GE.9).AND.(A(4).LE.11),ONE:
      IF,(A(4).GE.12).AND.(A(4).LE.14),TWO:
      IF,(A(4).GE.15).AND.(A(4).LE.18),THR:
      IF,(A(4).GE.1).AND.(A(4).LE.3),FOUR:
      ELSE,FIVE;
ONE   ASSIGN:J=0;
      BRANCH,1:ALWAYS,DEST;
TWO   ASSIGN:J=1;
      BRANCH,1:ALWAYS,DEST;
THR   ASSIGN:J=2;
      BRANCH,1:ALWAYS,DEST;
FOUR  ASSIGN:J=3;
      BRANCH,1:ALWAYS,DEST;
FIVE  ASSIGN:J=4;
      BRANCH,1:ALWAYS,DEST;
DEST  BRANCH,1:
      IF,(A(5).GE.25).AND.(A(5).LE.28),CIN1:
      IF,(A(5).GE.29).AND.(A(5).LE.31),CIN2:
      IF,(A(5).GE.32).AND.(A(5).LE.34),CIN3:
      IF,(A(5).GE.19).AND.(A(5).LE.21),CIN4:
      ELSE,CIN5;
CIN1  ASSIGN:A(11)=J*5+1;
      ROUTE:A(2)/A(8),SEQ;
CIN2  ASSIGN:A(11)=J*5+2;
      ROUTE:A(2)/A(8),SEQ;
CIN3  ASSIGN:A(11)=J*5+3;

```

```
ROUTE:A(2)/A(8),SEQ;  
CIN4  ASSIGN:A(11)=J*5+4;  
ROUTE:A(2)/A(8),SEQ;  
CIN5  ASSIGN:A(11)=J*5+5;  
ROUTE:A(2)/A(8),SEQ;  
END;
```

```

      SUBROUTINE EVENT(JOB,I)
C
C*****
C      SOHMRV.FOR
C      (CASE 1 - IMPROVED LAYOUT)
C
C      THIS SUBROUTINE DETERMINES WHICH EVENT OR WHICH ONE OF
C      THE FOLLOWING THREE SUBROUTINES WILL BE EXECUTED
C      THESE SUBROUTINES ARE USED FOR THE SHORTEST DISTANCE
C      ALGORITHM.
C*****
C
C      GOTO (1,2,3),I
C      1 CALL CINDEF(JOB)
C        RETURN
C      2 CALL RSDEF(JOB)
C        RETURN
C      3 CALL SHORDDST(JOB)
C        END
C
C
C      SUBROUTINE CINDEF(JOB)
C*****
C      THIS SUBROUTINE ASSIGN THE CINEMA SYMBOL, ACCORDING TO
C      THE STATION ORIGIN AND THE DESTINATION OF THE JOB, TO
C      ATTRIBUTE (11) OF THE JOB.
C
C      ORIG = STATION ORIGIN (STORED IN ATTRIBUTE # 4)
C      DEST = DESTINATION (STORED IN ATTRIBUTE # 5)
C      CATT = ZONE FOR THE STATION ORIGIN
C      DATT = ZONE FOR THE DESTINATION
C      VAL = THE CINEMA SYMBOL
C*****
C
C      COMMON/SIM/D(50),DL(50),S(50),SL(50),X(50),DTNOW,TNOW,
C      1ITFIN,J,NRUN
C      ORIG=A(JOB,4)
C      DEST=A(JOB,5)
C      IF ((ORIG.GE.9).AND.(ORIG.LE.11)) THEN
C        DATT=0.0
C      ELSEIF ((ORIG.GE.12).AND.(ORIG.LE.14)) THEN
C        DATT=1.0
C      ELSEIF ((ORIG.GE.15).AND.(ORIG.LE.18)) THEN
C        DATT=2.0
C      ELSEIF ((ORIG.GE.1).AND.(ORIG.LE.3)) THEN
C        DATT=3.0
C      ELSE
C        DATT=4.0

```

```

ENDIF
IF ((DEST.GE.25).AND.(DEST.LE.28)) THEN
    VAL=(DATT*5.0)+1.0
ELSEIF ((DEST.GE.29).AND.(DEST.LE.31)) THEN
    VAL=(DATT*5.0)+2.0
ELSEIF ((DEST.GE.32).AND.(DEST.LE.34)) THEN
    VAL=(DATT*5.0)+3.0
ELSEIF ((DEST.GE.19).AND.(DEST.LE.21)) THEN
    VAL=(DATT*5.0)+4.0
ELSE
    VAL=(DATT*5.0)+5.0
ENDIF
CALL SETA(JOB,11,VAL)
RETURN
END

C
C
SUBROUTINE RSDEF(JOB)
C*****
C    THIS SUBROUTINE ASSIGN THE SHORTEST ROUTE THE AGV NEEDS
C    TO TAKE TO GET TO ITS DESTINATION AT EACH INTERSECTION.
C
C    IJUNCT  =  INTERSECTION NUMBER (CURRENT LOCATION OF
C               THE AGV)
C    DEST    =  FINAL DESTINATION OF THE AGV
C    INUM    =  SEQUENCE (ROUTE) NUMBER
C    VAL     =  NEXT STATION TO BE SEIZE
C*****
C
COMMON/SIM/D(50),DL(50),S(50),SL(50),X(50),DTNOW,TNOW,
1TFIN,J,NRUN
IJUNCT=M(JOB)
DEST=A(JOB,5)
IF (IJUNCT.EQ.38) THEN
    INUM=38
    VAL=27
ELSEIF (IJUNCT.EQ.39) THEN
    IF ((DEST.EQ.13).OR.(DEST.EQ.30)) THEN
        INUM=40
        VAL=69
    ELSE
        INUM=39
        VAL=68
    ENDIF
ELSEIF (IJUNCT.EQ.40) THEN
    IF ((DEST.EQ.17).OR.(DEST.EQ.18)) THEN
        INUM=42
        VAL=74
    ELSEIF ((DEST.EQ.33).OR.(DEST.EQ.34)) THEN

```

```

        INUM=42
        VAL=74
    ELSE
        INUM=41
        VAL=75
    ENDIF
ELSEIF (IJUNCT.EQ.41) THEN
    IF ((DEST.EQ.3).OR.(DEST.EQ.21)) THEN
        INUM=44
        VAL=70
    ELSE
        INUM=43
        VAL=2
    ENDIF
ELSEIF (IJUNCT.EQ.42) THEN
    IF ((DEST.GE.5).AND.(DEST.LE.8)) THEN
        INUM=46
        VAL=5
    ELSE
        INUM=45
        VAL=83
    ENDIF
ELSEIF (IJUNCT.EQ.43) THEN
    IF ((DEST.GE.13).AND.(DEST.LE.18)) THEN
        INUM=48
        VAL=89
    ELSEIF ((DEST.GE.29).AND.(DEST.LE.34)) THEN
        INUM=48
        VAL=89
    ELSEIF ((DEST.GE.1).AND.(DEST.LE.3)) THEN
        INUM=49
        VAL=89
    ELSEIF ((DEST.GE.19).AND.(DEST.LE.21)) THEN
        INUM= 49
        VAL=89
    ELSEIF ((DEST.GE.4).AND.(DEST.LE.8)) THEN
        INUM=37
        VAL=89
    ELSEIF ((DEST.GE.22).AND.(DEST.LE.24)) THEN
        INUM=50
        VAL=89
    ELSE
        INUM=47
        VAL=99
    ENDIF
ENDIF
CALL SETNS(JOB,INUM)
CALL SETIS(JOB,0)
CALL SETA(JOB,1,VAL)
RETURN
END

```



```

C
C
      SUBROUTINE SHORTDST(JOB)
C*****
C
C      THIS SUBROUTINE WILL FIND THE CLOSEST REQUEST TO THE AGV.
C      THIS SUBROUTINE READS DATA FORM FILE "SOHMREV.DAT", WHICH
C      CONTAINS THE SEQUENCE OF INPUT STATIONS.
C
C      Z(K)      =  AN ARRAY THAT STORE THE INPUT STATION NUMBERS
C      TJOB      =  STATUS OF THE AGV:
C                   0 - PICKING UP A JOB
C                   3 - EMPTY (WITHOUT A REQUEST)
C      LASTDST   =  CURRENT LOCATION OF THE AGV
C      DEST      =  FINAL DESTINATION OF THE AGV
C      IONE      =  DROP-OFF STATION
C      ITWO      =  ZERO, A CHECK TO SEE IF THE LINE READ (IN
C                   SOHMREV.DAT) IS CORRECT.
C*****
C
      COMMON/SIM/D(50),DL(50),S(50),SL(50),X(50),DTNOW,TNOW,
      1TFIN,J,NRUN
      DIMENSION Z(18)
      OPEN(10,FILE='SOHMREV.DAT')
      REWIND 10
      LASTDST=A(JOB,4)
      TJOB=0
33  READ(10,*) IONE,ITWO
      IF ((IONE.EQ.LASTDST).AND.(ITWO.EQ.0)) THEN
        READ(10,*) Z(1),Z(2),Z(3),Z(4),Z(5),Z(6),Z(7),Z(8),Z(9),Z(10),
        READ(10,*) Z(11),Z(12),Z(13),Z(14),Z(15),Z(16),Z(17),Z(18)
      ELSE
        READ(10,*) IONE,ITWO
        READ(10,*) IONE,ITWO
        GOTO 33
      ENDIF
C
C      CHECK IF THERE IS ANY REQUEST AT THE CLOSEST INPUT
C      STATION. IF THERE IS NONE, CHECK AT THE NEXT CLOSEST
C      INPUT STATIONS. IF THERE IS NO MORE REQUEST, SEND THE
C      AGV TO THE PARKING LOT (STATION 35).
C
      DEST=35
      DO 1 K=1,18
        N=Z(K)
        IF ((NQ(N).NE.0).AND.(DEST.EQ.35)) DEST=N
1  CONTINUE
      IF (DEST.EQ.35) TJOB=3
      CALL SETA(JOB,5,DEST)
      CALL SETA(JOB,6,TJOB)

```

**RETURN**  
**END**

\*\*\*\*\*

SOHMREV.DAT  
(CASE 1 - IMPROVED LAYOUT)

THIS FILE CONTAINS THE DATA TO FIND THE CLOSEST  
STATION WITH REQUEST. THE ORGANIZATION OF THE DATA IS AS  
FOLLOWS: LINE 1 : DROP-OFF OR INTERSECTION STATION, ZERO  
LINE 2 : THE FIRST TEN CLOSEST INPUT STATIONS  
LINE 3 : THE SECOND EIGHT CLOSEST INPUT STATIONS

\*\*\*\*\*

19 0  
1 2 3 4 5 6 7 8 9 10  
13 14 11 15 12 16 17 18  
20 0  
2 3 4 5 6 7 8 9 10 13  
14 11 15 1 12 16 17 18  
21 0  
4 5 6 7 8 9 10 13 14 11  
15 1 12 16 2 17 3 18  
22 0  
9 10 4 13 5 14 11 6 7 8  
15 1 12 16 2 17 3 18  
23 0  
9 10 4 13 5 14 11 6 7 8  
15 1 12 16 2 17 3 18  
24 0  
9 10 4 13 5 14 11 6 7 8  
15 1 12 16 2 17 3 18  
25 0  
10 11 12 13 14 15 16 1 17 18  
2 3 4 5 6 7 8 9  
26 0  
10 11 12 13 14 15 16 1 17 18  
2 3 4 5 6 7 8 9  
27 0  
11 12 13 14 15 16 1 17 18 2  
3 4 5 6 7 8 9 10  
28 0  
11 12 13 14 15 16 1 17 18 2  
3 4 5 6 7 8 9 10  
29 0  
14 15 16 1 17 18 2 3 4 5  
6 7 8 9 10 13 11 12  
30 0  
14 15 16 1 17 18 2 3 4 5  
6 7 8 9 10 13 11 12  
31 0  
15 16 1 17 18 2 3 4 5 6  
7 8 9 10 13 14 11 12  
32 0

16 1 17 18 2 3 4 5 6 7  
 8 9 10 13 14 11 15 12  
 33 0  
 17 18 1 2 3 4 5 6 7 8  
 9 10 13 14 11 15 12 16  
 34 0  
 18 2 3 4 5 6 7 8 9 10  
 13 14 11 15 1 12 16 17  
 35 0  
 9 10 11 12 13 14 15 16 1 17  
 18 2 3 4 5 6 7 8  
 39 0  
 13 14 15 16 1 17 18 2 3 4  
 5 6 7 8 9 10 11 12  
 40 0  
 16 1 17 18 2 3 4 5 6 7  
 8 9 10 13 14 11 15 12  
 41 0  
 2 3 4 5 6 7 8 9 10 13  
 14 11 15 1 12 16 17 18  
 42 0  
 5 6 7 8 9 10 13 14 11 15  
 1 12 16 2 17 3 18 4  
 43 0  
 9 10 4 13 5 14 11 6 7 8  
 15 1 12 16 2 17 3 18

```

BEGIN;
;
;               OHMREV.EXP
;       ( CASE 1 - IMPROVED LAYOUT)
;
;       GIVE THE TITLE OF THE SUMMARY GENERATED AT THE END OF
;       SIMULATION RUN.
;
PROJECT,CASE 1 REVISED Q D,I. OETOMO,11/26/87;
;
;       DEFINE THE MAX. NUMBER OF CONCURRENT ENTITIES,
;       ATTRIBUTES, QUEUES, AND THE ATTRIBUTE NUMBER FOR CINEMA
;       SYMBOL.
;
DISCRETE,340,13,165,123,11;
;
;       INITIALIZE THE SEED VALUE FOR THE RANDOM NUMBER GENERATOR
;       STREAM (TWO SETS OF SEED VALUES: 3000, 5555 & 7777, 1000,
;       ARE USED TO SIMULATE EACH NUMBER OF AGVS), AND DEFINE ALL
;       PARAMETERS VALUES USED IN THE MODEL FILE.
;
SEEDS:1,3000:2,5555;
PARAMETERS:
1, .27,21,.33,23,.47,25,.6,26,.73,28,.97,29,1.,30:
2, .35,19,.45,22,1.,23:
3, .81,19,.83,20,.84,22,.85,25,.99,26,1.,27:
4, .99,19,1.,20:5,.99,19,1.0,29:6,1.,19:7,.99,19,1.,29:
8, .11,19,.12,20,.15,21,.38,22,.55,23,.57,26,.68,29,.7,30,
   .97,31,1.,33:
9, .17,19,.24,21,.26,22,.32,23,.4,26,.5,29,.51,30,.56,31,
   .96,33,1.,34:
10, .04,19,.06,21,.26,22,.48,24,.54,29,.55,30,.99,33,1.,34:
11, .59,22,.76,23,.82,29,1.,31:
12, .21,19,.25,20,.25,21,.42,22,.65,23,.72,26,.81,29,.83,30,
   .99,32,1.,33:
13, .33,19,.36,21,.46,22,.55,23,.56,24,.62,25,.7,26,.94,28,
   .96,31,1.,33:
14, .8,19,.81,21,.83,22,.99,23,1.,26:15,1.,19:
16, .05,20,.21,21,.44,22,.64,23,.65,24,.68,25,.73,26,.75,27,
   .76,28,.88,29,.9,30,.92,31,.94,32,.995,33,1.,34:
17, .44,19,.46,22,.71,25,.72,26,.94,28,.98,29,.99,30,1.,31:
18, .2,19,.4,25,.6,27,.8,28,1.,29:19,60,120:20,30,60:
21,8640,10560:22,2541,3106:23,1296,1584:24,762,932:
25,2400,2933:26,2700,3300:27,2400,2933:28,566,692:
29,939,1148:30,2356,2880:31,1194,1460:32,1906,2329:
33,1865,2072:34,456,507:35,9257,11314:36,247,302:
37,1464,1790:38,25920,28800;
;
;       DEFINE THE ROUTE FROM EACH INPUT, DROP-OFF STATIONS, AND
;       INTERSECTIONS (SEE EXPLANATION IN OHM.EXP FILE).
;

```

SEQUENCES:

- 1,20,78,24/78,101,28/101,41,29/41,,29:
- 2,46,80,12/80,81,16/81,82,30/82,100,30/100,102,30/102,4,30/  
4,103,20/103,42,31/42,,31:
- 3,21,55,10/55,80,44/80,81,16/81,82,30/82,100,30/100,102,30/  
102,4,30/4,103,20/103,42,31/42,,31:
- 4,103,42,31/42,,31:
- 5,6,85,30/85,86,6/86,56,32/56,48,40/48,87,20/87,22,12/  
22,104,16/104,23,25/23,37,25/37,24,30/24,43,30/43,,8:
- 6,85,86,6/86,56,32/56,48,40/48,87,20/87,22,12/22,104,16/  
104,23,25/23,37,25/37,24,30/24,43,30/43,,8:
- 7,8,57,30/57,47,39/47,48,39/48,87,20/87,22,12/22,104,16/  
104,23,25/23,37,25/37,24,30/24,43,30/43,,8:
- 8,57,47,39/47,48,39/48,87,20/87,22,12/22,104,16/104,23,25/  
23,37,25/37,24,30/24,43,30/43,,8:
- 9,64,105,34/105,25,34/25,106,34/106,26,31/26,10,31/  
10,107,24/107,38,34/38,,34:
- 10,107,38,34/38,,34:
- 11,65,66,38/66,108,38/108,109,38/109,12,39/12,49,39/49,67,20/  
67,39,24/39,,32:
- 12,49,67,20/67,39,24/39,,32:
- 13,30,59,56/59,71,24/71,110,12/110,14,20/14,111,20/111,31,20/  
31,73,20/73,112,31/112,15,31/15,32,32/32,36,12/36,117,24/  
117,40,24/40,,24:
- 14,111,31,20/31,73,20/73,112,31/112,15,31/15,32,32/32,36,12/  
36,117,24/117,40,24/40,,24:
- 15,32,36,12/36,117,24/117,40,24/40,,24:
- 16,76,118,12/118,52,12/52,77,12/77,19,18/19,1,18/1,20,12/  
20,78,24/78,101,28/101,41,29/41,,29:
- 17,34,18,56/18,60,36/60,76,6/76,118,12/118,52,12/52,77,12/  
77,19,18/19,1,18/1,20,12/20,78,24/78,101,28/101,41,29/41,,29:
- 18,60,76,6/76,118,12/118,52,12/52,77,12/77,19,18/19,1,18/  
1,20,12/20,78,24/78,101,28/101,41,29/41,,29:
- 19,1,20,12/20,78,24/78,101,28/101,41,29/41,,29:
- 20,78,101,28/101,41,29/41,,29:
- 21,55,80,44/80,81,16/81,82,30/82,100,30/100,102,30/102,4,30/  
4,103,20/103,42,31/42,,31:
- 22,104,23,25/23,37,25/37,24,30/24,43,30/43,,8:
- 23,37,24,30/24,43,30/43,,8:
- 24,43,,8:
- 25,106,26,31/26,10,31/10,107,24/107,38,34/38,,34:
- 26,10,107,24/107,38,34/38,,34:
- 27,28,11,4/11,65,32/65,66,38/66,108,38/108,109,38/109,12,39/  
12,49,39/49,67,20/67,39,24/39,,32:
- 28,11,65,32/65,66,38/66,108,38/108,109,38/109,12,39/12,49,39/  
49,67,20/67,39,24/39,,32:
- 29,71,110,12/110,14,20/14,111,20/111,31,20/31,73,20/73,112,31/  
112,15,31/15,32,32/32,36,12/36,117,24/117,40,24/40,,24:
- 30,59,71,24/71,110,12/110,14,20/14,111,20/111,31,20/31,73,20/  
73,112,31/112,15,31/15,32,32/32,36,12/36,117,24/117,40,24/  
40,,24:

```

31,73,112,31/112,15,31/15,32,32/32,36,12/36,117,24/117,40,24/
40,,24:
32,36,117,24/117,40,24/40,,24:
33,17,34,22/34,18,56/18,60,36/60,76,6/76,118,12/118,52,12/
52,77,12/77,19,18/19,1,18/1,20,12/20,78,24/78,101,28/
101,41,29/41,,29:
34,18,60,36/60,76,6/76,118,12/118,52,12/52,77,12/77,19,18/
19,1,18/1,20,12/20,78,24/78,101,28/101,41,29/41,,29:
35,9,64,41/64,105,34/105,25,34/25,106,34/106,26,31/26,10,31/
10,107,24/107,38,34/38,,34:
36,1,1,1:
37,89,90,24/90,114,40/114,91,40/91,92,40/92,94,40/94,53,40/
53,54,38/54,62,38/62,4,38/4,103,20/103,42,31/42,,31:
38,27,28,52/28,11,4/11,65,32/65,66,38/66,108,38/108,109,38/
109,12,39/12,49,39/49,67,20/67,39,24/39,,32:
39,68,29,12/29,71,24/71,110,12/110,14,20/14,111,20/111,31,20/
31,73,20/73,112,31/112,15,31/15,32,32/32,36,12/36,117,24/
117,40,24/40,,24:
40,69,13,12/13,30,38/30,59,56:
41,75,16,12/16,76,30/76,118,12/118,52,12/52,77,12/77,19,18/
19,1,18/1,20,12/20,78,24/78,101,28/101,41,29/41,,29:
42,74,113,24/113,33,39/33,17,39/17,34,22/34,18,56/18,60,36:
43,2,46,12/46,80,12/80,81,16/81,82,30/82,100,30/100,102,30/
102,4,30/4,103,20/103,42,31/42,,31:
44,70,79,22/79,3,40/3,21,40/21,55,10:
45,83,84,34/84,51,40/51,87,40/87,22,12/22,104,16/104,23,25/
23,37,25/37,24,30/24,43,30/43,,8:
46,5,6,26/6,85,30/85,88,6/88,7,12/7,8,12/8,57,30/57,47,39/
47,48,39/48,87,20/87,22,12/22,104,16/104,23,25/23,37,25/
37,24,30/24,43,30/43,,8:
47,99,35,25/35,9,41/9,64,41/64,105,34/105,25,34/25,106,34/
106,26,31/26,10,31/10,107,24/107,38,34/38,,34:
48,89,90,24/90,114,40/114,91,40/91,92,40/92,93,34/93,72,40/
72,58,34/58,67,34/67,39,24/39,,32:
49,89,90,24/90,114,40/114,91,40/91,92,40/92,94,40/94,95,40/
95,96,39/96,116,39/116,97,37/97,63,39/63,115,39/115,98,39/
98,61,30/61,77,30/77,19,18/19,1,18/1,20,12/20,78,24/
78,101,28/101,41,29/41,,29;
;
;   INITIALIZE ALL VARIABLES USED TO GATHER PERCENT LOADED,
;   UNLOADED TRAVEL, LOADING/UNLOADING, PARKING, AND WAITING
;   TIME
;
INITIALIZE,X(1)=0,X(2)=0,X(3)=0,X(4)=0,X(5)=0,X(6)=0,X(7)=0,
X(8)=0,X(9)=0,X(10)=0,X(11)=0,X(12)=0,X(13)=0,X(14)=0,
X(15)=0,X(16)=0,X(17)=0,X(18)=0,X(19)=0,X(20)=0,X(21)=0,
X(22)=0,X(23)=0,X(24)=0,X(25)=0,X(26)=0,X(27)=0,X(28)=0,
X(29)=0,X(30)=0,X(31)=0,X(32)=0,X(33)=0,X(34)=0,X(35)=0,
X(36)=0,X(37)=0,X(38)=0,X(39)=0,X(40)=0,X(41)=0,X(42)=0,
X(43)=0,X(44)=0,X(45)=0,X(46)=0,X(47)=0,X(48)=0;
;

```

```

;      PRINT OUT THE DATA COLLECTED DURING THE SIMULATION RUN.
;
TALLIES:1,TIME IN SYSTEM;
COUNTERS:1,THROUGHPUT;
DSTAT:1,X(1),AGV 1 LOADED:2,X(2),AGV 2 LOADED:3,X(3),AGV 3 LOADED:
4,X(4),AGV 4 LOADED:5,X(5),AGV 5 LOADED:6,X(6),AGV 6 LOADED:
7,X(7),AGV 7 LOADED:8,X(8),AGV 8 LOADED:9,X(9),AGV 9 LOADED:
10,X(10),AGV 10 LOADED:11,X(11),AGV 11 LOADED:
12,X(12),AGV 1 UNLOADED:13,X(13),AGV 2 UNLOADED:
14,X(14),AGV 3 UNLOADED:15,X(15),AGV 4 UNLOADED:
16,X(16),AGV 5 UNLOADED:17,X(17),AGV 6 UNLOADED:
18,X(18),AGV 7 UNLOADED:19,X(19),AGV 8 UNLOADED:
20,X(20),AGV 9 UNLOADED:21,X(21),AGV 10 UNLOADED:
22,X(22),AGV 11 UNLOADED:23,X(23),AGV 1 WAITING:
24,X(24),AGV 2 WAITING:25,X(25),AGV 3 WAITING:
26,X(26),AGV 4 WAITING:27,X(27),AGV 5 WAITING:
28,X(28),AGV 6 WAITING:29,X(29),AGV 7 WAITING:
30,X(30),AGV 8 WAITING:31,X(31),AGV 9 WAITING:
32,X(32),AGV 10 WAITING:33,X(33),AGV 11 WAITING:
34,X(34),AGV 1 LOADING:35,X(35),AGV 2 LOADING:
36,X(36),AGV 3 LOADING:37,X(37),AGV 4 LOADING:
38,X(38),AGV 5 LOADING:39,X(39),AGV 6 LOADING:
40,X(40),AGV 7 LOADING:41,X(41),AGV 8 LOADING:
42,X(42),AGV 9 LOADING:43,X(43),AGV 10 LOADING:
44,X(44),AGV 11 LOADING:45,X(47),GENERATED REQ:
46,X(48),ACCEPTED REQ;
;
;      SPECIFY THE NUMBER OF SEGMENTS IN THE SYSTEM AND THE
;      NUMBER AND LENGTH OF THE SIMULATION RUN(S).
;
RESOURCES:1-123,PATH;
REPLICA,1,0,28800;
END;

```



APPENDIX B  
PROGRAM FOR CASE 2

THM.MOD & THMREV.MOD

(CASE 2 BASIC & IMPROVED LAYOUT MODEL FILES)

Both SOHM.MOD and SOHMREV.MOD use 16 attributes: A(1) - A(16), and 2 variables: X(1) & X(2).

A(1) = Next segment to travel.  
A(2) = length of the segment to be traveled.  
A(3) = Current station number or segment just traveled.  
A(4) = Station of origin, also used as current location of AGV.  
A(5) = Destination of AGV or of part.  
A(6) = AGV status : 0 = picking up part  
                  1 = transporting part to destination  
                  3 = empty and going to parking lot.  
A(7) = parking time.  
A(8) = AGV speed.  
A(9) = origin of the AGV : 0 = from parking lot  
                          1 = not from parking lot.  
A(10) = part waiting time.  
A(11) = cinema attribute.  
A(12) = AGV number.  
A(13) = loaded time.  
A(14) = unloaded time.  
A(15) = waiting time.  
A(16) = loading/unloading time.  
  
X(1),X(2) = counters.



```

        BRANCH,1:ALWAYS,DCIN;
        CREATE,1,7:UN(39,2);
        ASSIGN:A(4)=14;
        BRANCH,1:ALWAYS,DCIN;
        CREATE,1,9:UN(40,2);
        ASSIGN:A(4)=15;
        BRANCH,1:ALWAYS,DCIN;
        CREATE,1,6:UN(41,2);
        ASSIGN:A(4)=16;
        BRANCH,1:ALWAYS,DCIN;
        CREATE,1,6:UN(42,2);
        ASSIGN:A(4)=17;
        BRANCH,1:ALWAYS,DCIN;
        CREATE,1,2:UN(43,2);
        ASSIGN:A(4)=18;
        BRANCH,1:ALWAYS,DCIN;
        CREATE,1,2:UN(44,2);
        ASSIGN:A(4)=19;
        BRANCH,1:ALWAYS,DCIN;
        CREATE,1,1:UN(45,2);
        ASSIGN:A(4)=20;
        BRANCH,1:ALWAYS,DCIN;
        CREATE,1,2:UN(46,2);
        ASSIGN:A(4)=21;
        BRANCH,1:ALWAYS,DCIN;
        CREATE,1,2:UN(47,2);
        ASSIGN:A(4)=22;
        BRANCH,1:ALWAYS,DCIN;
        CREATE,1,2:UN(48,2);
        ASSIGN:A(4)=23;
        BRANCH,1:ALWAYS,DCIN;
        CREATE,1,2:UN(49,2);
        ASSIGN:A(4)=24;
;
;   LIMIT THE TOTAL CONCURRENT JOBS IN THE SYSTEM TO 250.
;
DCIN   ASSIGN:
        X(2)=X(2)+1;
        BRANCH,1:
            IF,X(2).GT.250,END:
            ELSE,REQ;
;
;   IF TOTAL CONCURRENT JOBS > 250, DISPOSE THE INCOMING
;   JOBS.
;
END    ASSIGN:
        X(2)=X(2)-1:DISPOSE;
;
;   REQUEST FOR AN AGV (SEND A SIGNAL). IF THERE IS AN
;   AVAILABLE AGV IN THE PARKING LOT, REQUEST THAT AGV.
;   OTHERWISE WAIT FOR THE NEXT AVAILABLE AGV.

```

```

;
REQ      SIGNAL:1:MARK(10);
        ASSIGN:M=A(4);
        ASSIGN:A(11)=38;
;
;      WAIT UNTIL AN AGV IS ALLOCATED.
;
        QUEUE,M;
        WAIT:M+50,1;
        ASSIGN:A(11)=39;
;
;      IF AN AGV HAS BEEN ALLOCATED, WAIT UNTIL IT REACHES THE
;      JOB.
;
        QUEUE,M+170;
        WAIT:M+105,1;
;
;      GATHER STATISTICS FOR THE TIME BETWEEN JOB CREATION AND
;      THE TIME THE JOB IS PICKED UP, AND DECREMENT THE TOTAL
;      CONCURRENT JOBS IN THE SYSTEM.
;
        ASSIGN:X(2)=X(2)-1;
        TALLY:1,INT(10):DISPOSE;
;
;      GENERATION OF AGVs. ALL AGVs ARE CREATED AT THE
;      BEGINNING OF THE SIMULATION.
;
        CREATE,1:0,21;
        ASSIGN:X(1)=X(1)+1;
        ASSIGN:A(12)=X(1);
;
;      WAIT FOR A REQUEST.
;
WAIT      ASSIGN:A(11)=40;
        QUEUE,25:MARK(7);
        WAIT:1,1;
;
;      IF THE SIGNAL FOR A REQUEST (JOB) IS ACCEPTED, FIND THE
;      JOB USING THE MAXIMUM QUEUE SIZE SELECTION RULE. IF
;      THE JOB IS FOUND, GET THE JOB. IF NOT, WAIT FOR ANOTHER
;      REQUEST.
;      NOTE: THIS BLOCK WILL BE SUBSTITUTED WITH A FORTRAN
;      SUBROUTINE (USING AN EVENT BLOCK) WHEN THE
;      SHORTEST DISTANCE ALGORITHM IS USED.
;
FIND      FINDJ,1,24:
        MAX(NQ(J));
        BRANCH,1:
        IF,NQ(J).EQ.0,WAIT:
        ELSE,OUT;
;

```

```

; IF THE JOB IS FOUND, SEND A SIGNAL TO THE JOB TO INDICATE
; THAT AN AGV HAS BEEN ALLOCATED. SET ON EMPTY TRAVEL
; TIME, AND GATHER STATISTICS FOR PARKING TIME.
;
OUT    ASSIGN:A(5)=J:MARK(14);
      TALLY:6,INT(7);
      ASSIGN:A(11)=37;
      SIGNAL:A(5)+50;
      ASSIGN:M=145;
;
; IF THERE IS ANOTHER AGV GOING OUT OF PARKING LOT, WAIT
; UNTIL THAT AGV GETS OUT OF PARKING LOT.
;
      QUEUE,195:MARK(15);
      SEIZE:PATH(145);
      TALLY:4,INT(15);
      ASSIGN:X(3)=1;
      DELAY:UN(50,1),M;
      ASSIGN:X(3)=0;
;
; INITIALIZE THE ROUTE, ASSIGN SPEED AND CINEMA SYMBOL FOR
; AN EMPTY AGV, AND SET ON EMPTY TRAVEL TIME.
;
      ASSIGN:NS=56;
      ASSIGN:A(1)=124;
      ASSIGN:A(9)=0;
      ASSIGN:IS=0;
      ASSIGN:A(8)=5.0;
      ASSIGN:A(6)=0;
      ASSIGN:A(11)=37;
      ROUTE:5,96;
;
; STATION (MACRO) --
; CHECK (1) IF THE AGV JUST GOT OUT OF PARKING LOT
;      (2) IF AN INTERSECTION IS REACHED. IF IT'S,
;      ASSIGN THE ROUTE TO DESTINATION.
;
      STATION,1-144;
      BRANCH,1:
        IF,(M.EQ.96).AND.(A(9).EQ.0),
        OPKR:
        IF,(M.GE.51).AND.(M.LE.56),NRT:
        ELSE,NXT0;
NXT0   ASSIGN:A(3)=M;
NXT2   ASSIGN:M=A(1);
;
; SET ON WAITING TIME, SEIZE THE NEXT SEGMENT. IF THE NEXT
; SEGMENT IS OCCUPIED, WAIT UNTIL IT'S FREED. OTHERWISE,
; SEIZE THE SEGMENT AND RELEASE THE OCCUPIED SEGMENT.
;
NBLK   QUEUE,M+25:MARK(15);

```

```

        SEIZE:PATH(M);
        TALLY:4,INT(15);
        RELEASE:PATH(A(3));
        ASSIGN:M=A(3);
;
;   CHECK IF (1) THE AGV JUST GOT OUT OF PARKING LOT
;           (2) THE AGV IS EMPTY (WITHOUT A REQUEST)
;           (3) THE DESTINATION IS REACHED
;   IF NOT (1), (2), OR (3), PROCEED TO NEXT STATION.
;
        BRANCH,1:
            IF,A(9).EQ.0,STA:
            IF,A(6).EQ.3,CHK:
            IF,A(5).EQ.A(3),ARR:
            ELSE,GO;
STA      ASSIGN:M=96;
GO        ASSIGN:A(9)=1;
          ROUTE:A(2)/A(8),SEQ;
;
;   ARRIVAL -- COLLECT STATISTICS FOR EITHER EMPTY OR LOADED
;               TRAVEL, SET ON UNLOADING/LOADING TIME, AND
;               PICK UP OR DROP OFF THE JOB.
;
ARR      BRANCH,1:
          IF,A(6).EQ.1,I13:
          ELSE,I14;
I13      TALLY:2,INT(13);
          BRANCH,1:ALWAYS,DEL;
I14      TALLY:3,INT(14);
DEL      DELAY:UN(25,1),M:MARK(16);
          TALLY:5,INT(16);
          ASSIGN:NS=A(5);
          ASSIGN:IS=0;
          BRANCH,1:
            IF,A(6).EQ.1,DROP:
            ELSE,DELV;
;
;   DELIVERY -- SET ON LOADED TRAVEL TIME, AND ASSIGN THE
;               DESTINATION OF THE JOB ACCORDING TO THE
;               GIVEN PROBABILITY (FROM THE DATA).
;
DELV     ASSIGN:A(8)=4.0:MARK(13);
          ASSIGN:A(6)=1;
          ASSIGN:A(4)=A(5);
          ASSIGN:J=A(4)+105;
          SIGNAL:J;
          ASSIGN:A(5)=DP(A(4),1);
;
;   ASSIGN CINEMA SYMBOL ACCORDING TO THE STATION OF ORIGIN
;   AND THE DESTINATION OF THE JOB.
;

```

```

;
;   BRANCH,1:ALWAYS,CNM;
;
;   DROP-OFF -- SET ON EMPTY TRAVEL TIME, AND CHECK IF THERE
;               IS ANYMORE REQUEST.
;
DROP    ASSIGN:A(8)=5.0:MARK(14);
        ASSIGN:A(6)=0;
        ASSIGN:A(11)=37;
        COUNT:1;
;
;   FIND ANOTHER JOB. IF THERE IS NONE, SEND THE AGV TO THE
;   PARKING LOT AND CHECK FOR MORE JOB ON THE WAY TO PARKING
;   LOT. IF PARKING LOT IS REACHED, PARK THE AGV. IF NOT,
;   SEND THE AGV TO THE NEXT STATION.
;   NOTE: THE FOLLOWING BLOCKS WILL BE SUBSTITUTED WITH A
;   FORTRAN SUBROUTINE (USING AN EVENT BLOCK) WHEN
;   THE SHORTEST DISTANCE ALGORITHM IS USED TO FIND
;   THE NEXT JOB.
;
CHK      FINDJ,1,24:MAX(NQ(J));
        BRANCH,1:
            IF,(NQ(J).EQ.0).AND.(M.EQ.96),S145:
            IF,NQ(J).EQ.0,PKR:
            ELSE,PORD;
;
;   PARKING STATION -- WAIT IF THERE IS ANOTHER AGV IS
;   PARKING, AND SET OFF UNLOADED TRAVEL TIME.
;
S145     QUEUE,196:MARK(15);
        SEIZE:PATH(146);
        TALLY:4,INT(15);
        RELEASE:PATH(17);
        ASSIGN:M=96;
        ROUTE:5,146;
        STATION,146;
        RELEASE:PATH(146);
        ASSIGN:X(4)=1;
        DELAY:UN(50,1),M;
        ASSIGN:X(4)=0;
        TALLY:3,INT(14):MARK(7);
        BRANCH,1:ALWAYS,FIND;
;
;   IF ANOTHER JOB IS FOUND, PICK UP THE JOB.
;
PORD     ASSIGN:A(5)=J;
        SIGNAL:A(5)+50;
        ASSIGN:A(6)=0;
        BRANCH,1:
            IF,M.EQ.A(5),ARR:
            ELSE,GO;
;

```



```

;      IF NO MORE JOB IS FOUND, SEND THE AGV TO PARKING LOT.
;
PKR      ASSIGN:A(5)=17;
          ASSIGN:A(6)=3;
          ASSIGN:A(11)=39;
          ROUTE:A(2)/A(8),SEQ;
OPKR     ASSIGN:A(3)=145;
          BRANCH,1:ALWAYS,NXT2;
;
;      THE FOLLOWING BLOCKS ASSIGN THE SHORTEST ROUTE TO THE
;      DESTINATION (FROM AN INTERSECTION).
;      NOTE: THESE BLOCKS WILL BE SUBSTITUTED WITH A FORTRAN
;            SUBROUTINE (USING AN EVENT BLOCK) WHEN THE
;            SHORTEST DISTANCE ALGORITHM IS USED.
;
NRT      ASSIGN:IS=0;
          BRANCH,1:
              IF,M.EQ.51,C51:
              IF,M.EQ.52,C52:
              IF,M.EQ.53,C53:
              IF,M.EQ.54,C54:
              IF,M.EQ.55,C55:
              IF,M.EQ.56,C55;
C51      ASSIGN:NS=51;
          ASSIGN:A(1)=25;
          BRANCH,1:
              IF,(A(5).LE.6).AND.(A(5).NE.3),J51A:
              IF,(A(5).GE.27).AND.(A(5).LE.31),J51A:
              IF,(A(5).GE.18).AND.(A(5).LE.24),J51A:
              IF,(A(5).EQ.46),J51A:
              ELSE,NXT0;
J51A     ASSIGN:NS=52;
          ASSIGN:A(1)=138;
          BRANCH,1:ALWAYS,NXT0;
C52      ASSIGN:NS=53;
          ASSIGN:A(1)=140;
          BRANCH,1:
              IF,(A(5).EQ.32).OR.(A(5).EQ.9),J52A:
              IF,(A(5).GE.7).AND.(A(5).LE.17),J52B:
              IF,(A(5).GE.33).AND.(A(5).LE.43),J52B:
              ELSE,NXT0;
J52A     ASSIGN:NS=55;
          ASSIGN:A(1)=99;
          BRANCH,1:ALWAYS,NXT0;
J52B     ASSIGN:NS=54;
          ASSIGN:A(1)=99;
          BRANCH,1:ALWAYS,NXT0;
C53      ASSIGN:NS=50;
          ASSIGN:A(1)=36;
          BRANCH,1:
              IF,(A(5).GE.14).AND.(A(5).LE.17),J53A:

```

```

                IF, (A(5).GE.40).AND.(A(5).LE.43), J53A:
                ELSE, NXT0;
J53A    ASSIGN: NS=49;
        ASSIGN: A(1)=81;
        BRANCH, 1: ALWAYS, NXT0;
C54    ASSIGN: NS=48;
        ASSIGN: A(1)=82;
        BRANCH, 1:
            IF, (A(5).EQ.20).OR.(A(5).EQ.46), J54A:
            IF, (A(5).EQ.23).OR.(A(5).EQ.24), J54A:
            IF, (A(5).EQ.1).OR.(A(5).EQ.6), J54A:
            IF, (A(5).EQ.31).OR.(A(5).EQ.30), J54A:
            ELSE, NXT0;
J54A    ASSIGN: NS=47;
        ASSIGN: A(1)=113;
        BRANCH, 1: ALWAYS, NXT0;
C55    ASSIGN: NS=45;
        ASSIGN: A(1)=87;
        BRANCH, 1:
            IF, (A(5).GE.1).AND.(A(5).LE.17), J55A:
            IF, (A(5).GE.25).AND.(A(5).LE.43), J55A:
            ELSE, NXT0;
J55A    ASSIGN: NS=44;
        ASSIGN: A(1)=86;
        BRANCH, 1: ALWAYS, NXT0;
;
;   THE FOLLOWING BLOCKS ASSIGN THE CINEMA SYMBOL ACCORDING
;   TO THE STATION ORIGIN AND THE DESTINATION OF THE JOB.
;   NOTE: THESE BLOCKS WILL BE SUBSTITUTED WITH A FORTRAN
;         SUBROUTINE (AN EVENT BLOCK) WHEN THE SHORTEST
;         DISTANCE ALGORITHM IS USED.
;
CNM    BRANCH, 1:
        IF, (A(4).GE.14).AND.(A(4).LE.17), ONE:
        IF, (A(4).GE.10).AND.(A(4).LE.13), TWO:
        IF, (A(4).EQ.3).OR.(A(4).EQ.4), THR:
        IF, (A(4).GE.7).AND.(A(4).LE.9), THR:
        IF, (A(4).EQ.1).OR.(A(4).EQ.2), FOUR:
        IF, (A(4).EQ.5).OR.(A(4).EQ.6), FOUR:
        IF, (A(4).EQ.21).OR.(A(4).EQ.22), FIVE:
        IF, (A(4).EQ.18).OR.(A(4).EQ.19), FIVE:
        ELSE, SIX;
ONE    ASSIGN: J=0;
        BRANCH, 1: ALWAYS, DEST;
TWO    ASSIGN: J=1;
        BRANCH, 1: ALWAYS, DEST;
THR    ASSIGN: J=2;
        BRANCH, 1: ALWAYS, DEST;
FOUR   ASSIGN: J=3;
        BRANCH, 1: ALWAYS, DEST;
FIVE   ASSIGN: J=4;

```

```

        BRANCH, 1: ALWAYS, DEST;
SIX      ASSIGN: J=5;
        BRANCH, 1: ALWAYS, DEST;
DEST     BRANCH, 1:
          IF, (A(5).GE.40).AND.(A(5).LE.43), E11:
          IF, (A(5).GE.36).AND.(A(5).LE.39), E12:
          IF, (A(5).GE.25).AND.(A(5).LE.27), E13:
          IF, (A(5).GE.32).AND.(A(5).LE.35), E13:
          IF, (A(5).GE.28).AND.(A(5).LE.31), E14:
          IF, (A(5).EQ.18).OR.(A(5).EQ.19), E15:
          IF, (A(5).EQ.21).OR.(A(5).EQ.22), E15:
          ELSE, E16;
E11      ASSIGN: A(11)=J*6+1;
          ROUTE: A(2)/A(8), SEQ;
E12      ASSIGN: A(11)=J*6+2;
          ROUTE: A(2)/A(8), SEQ;
E13      ASSIGN: A(11)=J*6+3;
          ROUTE: A(2)/A(8), SEQ;
E14      ASSIGN: A(11)=J*6+4;
          ROUTE: A(2)/A(8), SEQ;
E15      ASSIGN: A(11)=J*6+5;
          ROUTE: A(2)/A(8), SEQ;
E16      ASSIGN: A(11)=J*6+6;
          ROUTE: A(2)/A(8), SEQ;
END;

```

```

      SUBROUTINE EVENT(JOB,I)
C*****
C          STHM.FOR
C          (CASE 2 - BASIC LAYOUT)
C
C      THIS SUBROUTINE DETERMINES WHICH EVENT OR WHICH ONE OF
C      THE FOLLOWING THREE SUBROUTINES WILL BE EXECUTED
C      THESE SUBROUTINES ARE USED FOR THE SHORTEST DISTANCE
C      ALGORITHM.
C*****
C
C      GOTO (1,2,3),I
1     CALL CINDEF(JOB)
      RETURN
2     CALL RSDEF(JOB)
      RETURN
3     CALL SHORSTDST(JOB)
      END
C
C
      SUBROUTINE CINDEF(JOB)
C*****
C      THIS SUBROUTINE ASSIGN THE CINEMA SYMBOL, ACCORDING TO
C      THE STATION ORIGIN AND THE DESTINATION OF THE JOB, TO
C      ATTRIBUTE (11) OF THE JOB.
C
C      ORIG = STATION ORIGIN (STORED IN ATTRIBUTE # 4)
C      DEST = DESTINATION (STORED IN ATTRIBUTE # 5)
C      CATT = ZONE FOR THE STATION ORIGIN
C      DATT = ZONE FOR THE DESTINATION
C      VAL  = THE CINEMA SYMBOL
C
C*****
C
      COMMON/SIM/D(50),DL(50),S(50),SL(50),X(50),DTNOW,TNOW,
1TFIN,J,NRUN
      ORIG=A(JOB,4)
      DEST=A(JOB,5)
      IF ((ORIG.GE.14).AND.(ORIG.LE.17)) THEN
          CATT=0.0
      ELSEIF ((ORIG.GE.10).AND.(ORIG.LE.13)) THEN
          CATT=1.0
      ELSEIF ((ORIG.EQ.3).OR.(ORIG.EQ.4)) THEN
          CATT=2.0
      ELSEIF ((ORIG.GE.7).AND.(ORIG.LE.9)) THEN
          CATT=2.0
      ELSEIF ((ORIG.EQ.1).OR.(ORIG.EQ.2)) THEN
          CATT=3.0
      ELSEIF ((ORIG.EQ.5).OR.(ORIG.EQ.6)) THEN

```

```

      CATT=3.0
    ELSEIF ((ORIG.EQ.21).OR.(ORIG.EQ.22)) THEN
      CATT=4.0
    ELSEIF ((ORIG.EQ.18).OR.(ORIG.EQ.19)) THEN
      CATT=4.0
    ELSE
      CATT=5.0
    ENDIF
    IF ((DEST.GE.40).AND.(DEST.LE.43)) THEN
      VAL=(CATT*6.0)+1.0
    ELSEIF ((DEST.GE.36).AND.(DEST.LE.39)) THEN
      VAL=(CATT*6.0)+2.0
    ELSEIF ((DEST.GE.25).AND.(DEST.LE.27)) THEN
      VAL=(CATT*6.0)+3.0
    ELSEIF ((DEST.GE.32).AND.(DEST.LE.35)) THEN
      VAL=(CATT*6.0)+3.0
    ELSEIF ((DEST.GE.28).AND.(DEST.LE.31)) THEN
      VAL=(CATT*6.0)+4.0
    ELSEIF ((DEST.EQ.18).OR.(DEST.EQ.19)) THEN
      VAL=(CATT*6.0)+5.0
    ELSEIF ((DEST.EQ.21).OR.(DEST.EQ.22)) THEN
      VAL=(CATT*6.0)+5.0
    ELSE
      VAL=(CATT*6.0)+6.0
    ENDIF
    CALL SETA(JOB,11,VAL)
    RETURN
  END

C
C
  SUBROUTINE RSDEF(JOB)
C*****
C
C   THIS SUBROUTINE ASSIGN THE SHORTEST ROUTE THE AGV NEEDS
C   TO TAKE TO GET TO ITS DESTINATION AT EACH INTERSECTION.
C
C   IJUNCT  =  INTERSECTION NUMBER (CURRENT LOCATION OF
C               THE AGV)
C   DEST    =  FINAL DESTINATION OF THE AGV
C   INUM    =  SEQUENCE (ROUTE) NUMBER
C   VAL     =  NEXT STATION TO BE SEIZE
C   JCT     =  ZONE FOR THE STATIONS
C
C*****
C
  COMMON/SIM/D(50),DL(50),S(50),SL(50),X(50),DTNOW,TNOW,
  1TFIN,J,NRUN
  IJUNCT=M(JOB)
  DEST=A(JOB,5)
  JCT=0
  IF (IJUNCT.EQ.51) THEN

```

```

      IF ((DEST.LE.6).AND.(DEST.NE.3)) JCT=1
      IF ((DEST.GE.27).AND.(DEST.LE.31)) JCT=1
      IF ((DEST.GE.18).AND.(DEST.LE.24)) JCT=1
      IF (DEST.EQ.46) JCT=1
      IF (JCT.EQ.0) THEN
        INUM=51
        VAL=25
      ELSE
        INUM=52
        VAL=138
      ENDIF
    ELSEIF (IJUNCT.EQ.52) THEN
      VAL=99
      IF ((DEST.EQ.32).OR.(DEST.EQ.9)) THEN
        INUM=55
      ELSEIF ((DEST.GE.7).AND.(DEST.LE.17)) THEN
        INUM=54
      ELSEIF ((DEST.GE.33).AND.(DEST.LE.43)) THEN
        INUM=54
      ELSE
        INUM=53
        VAL=140
      ENDIF
    ELSEIF (IJUNCT.EQ.53) THEN
      IF ((DEST.GE.14).AND.(DEST.LE.17)) THEN
        INUM=49
        VAL=81
      ELSEIF ((DEST.GE.40).AND.(DEST.LE.43)) THEN
        INUM=49
        VAL=81
      ELSE
        INUM=50
        VAL=36
      ENDIF
    ELSEIF (IJUNCT.EQ.54) THEN
      IF ((DEST.EQ.20).OR.(DEST.EQ.46)) JCT=1
      IF ((DEST.EQ.23).OR.(DEST.EQ.24)) JCT=1
      IF ((DEST.EQ.1).OR.(DEST.EQ.6)) JCT=1
      IF ((DEST.EQ.31).OR.(DEST.EQ.30)) JCT=1
      IF (JCT.EQ.1) THEN
        INUM=47
        VAL=113
      ELSE
        INUM=48
        VAL=82
      ENDIF
    ELSEIF ((IJUNCT.EQ.55).OR.(IJUNCT.EQ.56)) THEN
      IF ((DEST.GE.1).AND.(DEST.LE.17)) THEN
        INUM=44
        VAL=86
      ELSEIF ((DEST.GE.25).AND.(DEST.LE.43)) THEN

```

```

        INUM=44
        VAL=86
    ELSE
        INUM=45
        VAL=87
    ENDIF
ENDIF
CALL SETNS(JOB, INUM)
CALL SETIS(JOB, 0)
CALL SETA(JOB, 1, VAL)
RETURN
END

C
C
SUBROUTINE SHORTDST(JOB)
C*****
C
C    THIS SUBROUTINE WILL FIND THE CLOSEST REQUEST TO THE AGV.
C    THIS SUBROUTINE READS DATA FORM FILE "STM.DAT", WHICH
C    CONTAINS THE SEQUENCE OF INPUT STATIONS.
C
C    Z(K)      =  AN ARRAY THAT STORE THE INPUT STATION NUMBERS
C    TJOB      =  STATUS OF THE AGV:
C                  0 - PICKING UP A JOB
C                  3 - EMPTY (WITHOUT A REQUEST)
C    LASTDST   =  CURRENT LOCATION OF THE AGV
C    DEST      =  FINAL DESTINATION OF THE AGV
C    IONE      =  DROP-OFF STATION NUMBER
C    ITWO      =  ZERO, A CHECK TO SEE IF THE LINE READ (IN
C                  STM.DAT) IS CORRECT.
C*****
C
COMMON/SIM/D(50),DL(50),S(50),SL(50),X(50),DTNOW,TNOW,
1TFIN,J,NRUN
DIMENSION Z(24)
OPEN(10,FILE='STM.DAT')
REWIND 10
LASTDST=A(JOB,4)
33 READ(10,*) IONE,ITWO
IF ((IONE.EQ.LASTDST).AND.(ITWO.EQ.0)) THEN
    READ(10,*) Z(1),Z(2),Z(3),Z(4),Z(5),Z(6),Z(7),Z(8)
    READ(10,*) Z(9),Z(10),Z(11),Z(12),Z(13),Z(14),Z(15),Z(16)
    READ(10,*) Z(17),Z(18),Z(19),Z(20),Z(21),Z(22),Z(23),Z(24)
ELSE
    READ(10,*) IONE,ITWO
    READ(10,*) IONE,ITWO
    READ(10,*) IONE,ITWO
    GOTO 33
ENDIF
C

```

```

C      CHECK IF THERE IS ANY REQUEST AT THE CLOSEST INPUT
C      STATION. IF THERE IS NONE, CHECK AT THE NEXT CLOSEST
C      INPUT STATIONS. IF THERE IS NO MORE REQUEST, SEND THE
C      AGV TO THE PARKING LOT (STATION 100).
C

```

```

      DEST=100
      TJOB=0
      DO 1 K=1,24
        N=Z(K)
        IF ((NQ(N).NE.0).AND.(DEST.EQ.100)) DEST=N
1     CONTINUE
      IF (DEST.EQ.100) THEN
        TJOB=3
        DEST=17
      ENDIF
      CALL SETA(JOB,5,DEST)
      CALL SETA(JOB,6,TJOB)
      RETURN
      END

```



\*\*\*\*\*

STM.DAT  
(CASE 2 - BASIC LAYOUT)

THIS FILE CONTAINS THE DATA TO FIND THE CLOSEST  
STATION WITH REQUEST. THE ORGANIZATION OF THE DATA IS AS  
FOLLOWS: LINE 1 : DROP-OFF OR INTERSECTION STATION, ZERO  
LINE 2 : THE FIRST EIGHT CLOSEST INPUT STATIONS  
LINE 3 : THE SECOND EIGHT CLOSEST INPUT STATIONS  
LINE 4 : THE LAST EIGHT CLOSEST INPUT STATIONS

\*\*\*\*\*

18 0  
19 20 22 21 24 23 2 18  
6 4 1 3 7 5 8 9  
10 11 12 14 15 16 13 17  
19 0  
20 22 21 24 23 2 18 6  
4 19 1 3 7 5 8 9  
10 11 12 14 15 16 13 17  
20 0  
24 23 6 1 2 18 4 19  
3 7 5 8 9 20 22 10  
11 12 21 14 15 16 13 17  
21 0  
2 18 4 19 3 7 5 8  
9 20 6 22 1 10 11 12  
21 24 14 15 23 16 13 17  
22 0  
21 2 18 4 19 3 7 5  
8 9 20 6 22 1 10 11  
12 24 14 15 23 16 13 17  
23 0  
6 1 2 18 4 19 3 7  
5 8 9 20 22 10 11 12  
21 24 14 15 23 16 13 17  
24 0  
23 6 1 2 18 4 19 3  
7 5 8 9 20 22 10 11  
12 21 24 14 15 16 13 17  
25 0  
3 7 8 9 5 10 11 12  
6 14 1 2 15 18 4 16  
13 19 17 20 22 21 24 23  
26 0  
3 5 6 1 2 18 4 19  
7 8 9 20 22 10 11 12  
21 24 14 15 23 16 13 17  
27 0  
5 6 1 2 18 4 19 3  
7 8 9 20 22 10 11 12

21 24 14 15 23 16 13 17  
 28 0  
 6 1 2 18 4 19 3 7  
 5 8 9 20 22 10 11 12  
 21 24 14 15 23 16 13 17  
 29 0  
 6 1 2 18 4 19 3 7  
 5 8 9 20 22 10 11 12  
 21 24 14 15 23 16 13 17  
 30 0  
 1 2 18 4 19 3 7 5  
 8 9 20 6 22 10 11 12  
 21 24 14 15 23 16 13 17  
 31 0  
 1 2 18 4 19 3 7 5  
 8 9 20 6 22 10 11 12  
 21 24 14 15 23 16 13 17  
 32 0  
 9 10 11 12 14 15 16 13  
 17 2 18 4 19 3 7 5  
 8 20 6 22 1 21 24 23  
 33 0  
 10 11 12 14 15 16 13 17  
 2 18 4 19 3 7 5 8  
 9 20 6 22 1 21 24 23  
 34 0  
 10 11 12 14 15 16 13 17  
 2 18 4 19 3 7 5 8  
 9 20 6 22 1 21 24 23  
 35 0  
 10 11 12 14 15 16 13 17  
 2 18 4 19 3 7 5 8  
 9 20 6 22 1 21 24 23  
 36 0  
 10 11 12 13 2 18 4 19  
 3 7 5 8 9 20 6 22  
 1 21 24 14 15 23 16 17  
 37 0  
 13 2 18 4 19 3 7 5  
 8 9 20 6 22 1 10 11  
 12 21 24 14 15 23 16 17  
 38 0  
 13 2 18 4 19 3 7 5  
 8 9 20 6 22 1 10 11  
 12 21 24 14 15 23 16 17  
 39 0  
 13 2 18 4 19 3 7 5  
 8 9 20 6 22 1 10 11  
 12 21 24 14 15 23 16 17  
 40 0  
 14 15 16 17 2 18 4 19

3 7 5 8 9 20 6 22  
 1 10 11 12 21 24 23 13  
 41 0  
 15 16 17 2 18 4 19 3  
 7 5 8 9 20 6 22 1  
 10 11 12 21 24 14 23 13  
 42 0  
 15 16 17 2 18 4 19 3  
 7 5 8 9 20 6 22 1  
 10 11 12 21 24 14 23 13  
 43 0  
 2 18 4 19 3 7 5 8  
 9 20 6 22 1 10 11 12  
 21 24 14 15 23 16 13 17  
 46 0  
 20 24 23 6 1 2 18 4  
 19 3 7 5 8 9 22 10  
 11 12 21 14 15 16 13 17  
 51 0  
 4 3 7 5 8 9 6 1  
 10 11 12 2 14 18 15 19  
 16 13 20 17 22 21 24 23  
 52 0  
 3 7 8 9 5 10 11 12  
 6 14 1 2 15 18 4 16  
 13 19 17 20 22 21 24 23  
 53 0  
 10 11 12 14 15 16 13 17  
 2 18 4 19 3 7 5 8  
 9 20 6 22 1 21 24 23  
 54 0  
 20 22 21 24 23 2 18 6  
 4 19 1 3 7 5 8 9  
 10 11 12 14 15 16 13 17  
 55 0  
 2 18 4 19 3 7 5 8  
 9 20 6 22 1 10 11 12  
 21 24 14 15 23 16 13 17  
 56 0  
 2 18 4 19 3 7 5 8  
 9 20 6 22 1 10 11 12  
 21 24 14 15 23 16 13 17  
 124 0  
 17 2 18 4 19 3 7 5  
 8 9 20 6 22 1 10 11  
 12 21 24 14 15 23 16 13

```

BEGIN;
;
;
;           THM.EXP
;       ( CASE 2 - BASIC LAYOUT)
;
;       GIVE THE TITLE OF THE SUMMARY GENERATED AT THE END OF
;       SIMULATION RUN.
;
PROJECT,CASE 2 BASIC Q D,I. OETOMO,12/20/87;
;
;       DEFINE THE MAX. NUMBER OF CONCURRENT ENTITIES,
;       ATTRIBUTES, QUEUES, AND THE ATTRIBUTE NUMBER FOR CINEMA
;       SYMBOL.
;
DISCRETE,300,16,197,150,11;
;
;       INITIALIZE THE SEED VALUE FOR THE RANDOM NUMBER GENERATOR
;       STREAM (TWO SETS OF SEED VALUES: 3000, 5555 & 7777, 1000,
;       ARE USED TO SIMULATE EACH NUMBER OF AGVs), AND DEFINE ALL
;       PARAMETERS VALUES USED IN THE MODEL FILE.
;
SEEDS:1,7777:2,1000;
PARAMETERS:
1,.014,28,.495,29,.569,18,.643,19,.704,46,.778,21,.852,22,
.926,23,1.,24:
2,.216,27,.34,18,.465,19,.502,46,.626,21,.751,22,.875,23,1.,24:
3,.631,43,.684,18,.736,19,.789,46,.842,21,.894,22,.947,23,1.,24:
4,.5,38,1.,39:
5,.073,30,.207,18,.341,19,.463,46,.597,21,.731,22,.865,23,1.,24:
6,.459,29,.466,43,.543,18,.62,19,.691,46,.768,21,.845,22,
.922,23,1,24:
7,.142,18,.285,19,.428,46,.571,21,.714,22,.857,23,1.,24:
8,.142,18,.285,19,.428,46,.571,21,.714,22,.857,23,1.,24:
9,.142,18,.285,19,.428,46,.571,21,.714,22,.857,23,1.,24:
10,.641,26,.692,18,.743,19,.794,46,.846,21,.897,22,.948,23,
1.,24:
11,.819,43,.844,18,.868,19,.901,46,.926,21,.948,22,.974,23,
1.,24:
12,.821,43,.844,18,.868,19,.901,46,.926,21,.950,22,.975,23,
1.,24:
13,.063,27,.898,31,.915,18,.932,19,.949,21,.966,22,.983,23,
1.,24:
14,.707,43,.75,18,.792,19,.830,46,.872,27,.915,22,.957,23,
1.,24:
15,1.,43:16,1.,43:17,1.,43:
18,.087,25,.158,26,.170,28,.175,29,.203,30,.333,31,.347,32,
.361,33,.368,34,.447,35,.462,36,.493,37,.672,38,.85,39,
.892,40,.921,41,.953,42,1.,46:
19,.087,25,.158,26,.170,28,.175,29,.203,30,.333,31,.347,32,
.361,33,.368,34,.447,35,.462,36,.493,37,.672,38,.85,39,
.892,40,.921,41,.953,42,1.,46:

```

```

20,.017,25,.038,26,.039,29,.059,30,.189,31,.193,32,.197,33,
  .199,34,.222,35,.226,36,.235,37,.260,38,.285,39,.298,40,
  .307,41,.316,42,.430,18,.544,19,.658,21,.772,22,.886,23,
  1.,24:
21,.087,25,.158,26,.170,28,.175,29,.203,30,.333,31,.347,32,
  .361,33,.368,34,.447,35,.462,36,.493,37,.672,38,.85,39,
  .892,40,.921,41,.953,42,1.,46:
22,.087,25,.158,26,.170,28,.175,29,.203,30,.333,31,.347,32,
  .361,33,.368,34,.447,35,.462,36,.493,37,.672,38,.85,39,
  .892,40,.921,41,.953,42,1.,46:
23,.087,25,.158,26,.170,28,.175,29,.203,30,.333,31,.347,32,
  .361,33,.368,34,.447,35,.462,36,.493,37,.672,38,.85,39,
  .892,40,.921,41,.953,42,1.,46:
24,.087,25,.158,26,.170,28,.175,29,.203,30,.333,31,.347,32,
  .361,33,.368,34,.447,35,.462,36,.493,37,.672,38,.85,39,
  .892,40,.921,41,.953,42,1.,46:
25,60,90:
26,202,247:27,355,433:28,1137,1389:29,2160,2640:30,3161,3863:
31,870,1063:32,18514,22629:33,12343,15086:34,12323,15086:
35,3323,4062:36,1062,1298:37,292,356:38,549,671:39,1223,1494:
40,1620,1980:41,1037,1267:42,1037,1267:43,392,479:44,392,479:
45,259,317:46,392,479:47,392,479:48,392,479:49,392,479:50,30,50;
;
;   DEFINE THE ROUTE FROM EACH INPUT, DROP-OFF STATIONS, AND
;   INTERSECTIONS (SEE EXPLANATION IN OHM.EXP FILE).
;
SEQUENCES:
1,56,,9:
2,108,59,40/59,51,39/51,,39:
3,92,133,36/133,60,38/60,88,38/88,5,29/5,67,29/67,28,52/
  28,29,52/29,63,10/63,85,30/85,6,21/6,30,4/30,31,9/
  31,109,12/109,1,28/1,56,28/56,,9:
4,27,134,21/134,71,22/71,88,22/88,5,29/5,67,29/67,28,52/
  28,29,52/29,63,10/63,85,30/85,6,21/6,30,4/30,31,9/
  31,109,12/109,1,28/1,56,28/56,,9:
5,67,28,52/28,29,52/29,63,10/63,85,30/85,6,21/6,30,4/
  30,31,9/31,109,12/109,1,28/1,56,28/56,,9:
6,30,31,9/31,109,12/109,1,28/1,56,28/56,,9:
7,68,8,42/8,33,42/33,34,14/34,73,2/73,91,30/91,47,30/
  47,35,17/35,53,17/53,,34:
8,33,34,14/34,73,2/73,91,30/91,47,30/47,35,17/35,53,17/
  53,,34:
9,62,91,90/91,47,30/47,35,17/35,53,17/53,,34:
10,11,12,16/12,69,16/69,37,32/37,95,32/95,77,44/77,38,44/
  38,94,44/94,78,45/78,80,45/80,39,45/39,13,44/13,119,28/
  119,76,24/76,89,24/89,70,21/70,55,22/55,,22:
11,12,69,16/69,37,32/37,95,32/95,77,44/77,38,44/38,94,44/
  94,78,45/78,80,45/80,39,45/39,13,44/13,119,28/119,76,24/
  76,89,24/89,70,21/70,55,22/55,,22:
12,69,37,32/37,95,32/95,77,44/77,38,44/38,94,44/94,78,45/
  78,80,45/80,39,45/39,13,44/13,119,28/119,76,24/76,89,24/

```

89, 70, 21/70, 55, 22/55, , 22:  
 13, 119, 76, 24/76, 89, 24/89, 70, 21/70, 55, 22/55, , 22:  
 14, 48, 41, 23/41, 121, 23/121, 42, 39/42, 15, 39/15, 93, 58/93, 118, 35/  
 118, 123, 35/123, 16, 35/16, 96, 35/96, 124, 40/124, 17, 40/17, 43, 40/  
 43, 131, 48/131, 64, 24/64, 90, 24/90, 65, 12/65, 89, 26/89, 70, 21/  
 70, 55, 22/55, , 22:  
 15, 93, 118, 35/118, 123, 35/123, 16, 35/16, 96, 35/96, 124, 40/  
 124, 17, 40/17, 43, 40/43, 131, 48/131, 64, 24/64, 90, 24/  
 90, 65, 12/65, 89, 26/89, 70, 21/70, 55, 22/55, , 22:  
 16, 96, 124, 40/124, 17, 40/17, 43, 40/43, 131, 48/131, 64, 24/64, 90, 24/  
 90, 65, 12/65, 89, 26/89, 70, 21/70, 55, 22/55, , 22:  
 17, 43, 131, 48/131, 64, 24/64, 90, 24/90, 65, 12/65, 89, 26/89, 70, 21/  
 70, 55, 22/55, , 22:  
 18, 102, 111, 45/111, 19, 45/19, 54, 44/54, , 41:  
 19, 54, , 41:  
 20, 104, 114, 40/114, 125, 40/125, 105, 40/105, 24, 39/24, 84, 39/  
 84, 106, 46/106, 116, 47/116, 126, 47/126, 23, 47/23, 103, 47/  
 103, 107, 50/107, 128, 50/128, 74, 49/74, 85, 49/85, 6, 12/6, 30, 4/  
 30, 31, 9/31, 109, 12/109, 1, 28/1, 56, 28/56, , 9:  
 21, 97, 122, 38/122, 75, 38/75, 90, 38/90, 65, 12/65, 89, 26/89, 70, 21/  
 70, 55, 22/55, , 22:  
 22, 98, 132, 44/132, 21, 45/21, 97, 45/97, 122, 38/122, 75, 38/75, 90, 38/  
 90, 65, 12/65, 89, 26/89, 70, 21/70, 55, 22/55, , 22:  
 23, 103, 107, 50/107, 128, 50/128, 74, 49/74, 85, 49/85, 6, 12/6, 30, 4/  
 30, 31, 9/31, 109, 12/109, 1, 28/1, 56, 28/56, , 9:  
 24, 84, 106, 46/106, 116, 47/116, 126, 47/126, 23, 47/23, 103, 47/  
 103, 107, 50/107, 128, 50/128, 74, 49/74, 85, 49/85, 6, 12/6, 30, 4/  
 30, 31, 9/31, 109, 12/109, 1, 28/1, 56, 28/56, , 9:  
 25, 52, , 50:  
 26, 3, 92, 21/92, 133, 36/133, 60, 38/60, 88, 38/88, 5, 29/5, 67, 29/  
 67, 28, 52/28, 29, 52/29, 63, 10/63, 85, 30/85, 6, 21/6, 30, 4/  
 30, 31, 9/31, 109, 12/109, 1, 28/1, 56, 28/56, , 9:  
 27, 134, 71, 22/71, 88, 22/88, 5, 29/5, 67, 29/67, 28, 52/28, 29, 52/  
 29, 63, 10/63, 85, 30/85, 6, 21/6, 30, 4/30, 31, 9/31, 109, 12/  
 109, 1, 28/1, 56, 28/56, , 9:  
 28, 29, 63, 10/63, 85, 30/85, 6, 21/6, 30, 4/30, 31, 9/31, 109, 12/  
 109, 1, 28/1, 56, 28/56, , 9:  
 29, 63, 85, 30/85, 6, 21/6, 30, 4/30, 31, 9/31, 109, 12/109, 1, 28/  
 1, 56, 28/56, , 9:  
 30, 31, 109, 12/109, 1, 28/1, 56, 28/56, , 9:  
 31, 109, 1, 28/1, 56, 28/56, , 9:  
 32, 9, 62, 40/62, 91, 90/91, 47, 30/47, 35, 17/35, 53, 17/53, , 34:  
 33, 34, 73, 2/73, 91, 30/91, 47, 30/47, 35, 17/35, 53, 17/53, , 34:  
 34, 73, 91, 30/91, 47, 30/47, 35, 17/35, 53, 17/53, , 34:  
 35, 53, , 34:  
 36, 10, 11, 10/11, 12, 16/12, 69, 16/69, 37, 32/37, 95, 32/95, 77, 44/  
 77, 38, 44/38, 94, 44/94, 78, 45/78, 80, 45/80, 39, 45/39, 13, 44/  
 13, 119, 28/119, 76, 24/76, 89, 24/89, 70, 21/70, 55, 22/55, , 22:  
 37, 95, 77, 44/77, 38, 44/38, 94, 44/94, 78, 45/78, 80, 45/80, 39, 45/  
 39, 13, 44/13, 119, 28/119, 76, 24/76, 89, 24/89, 70, 21/70, 55, 22/  
 55, , 22:

38, 94, 78, 45/78, 80, 45/80, 39, 45/39, 13, 44/13, 119, 28/119, 76, 24/  
 76, 89, 24/89, 70, 21/70, 55, 22/55, , 22:  
 39, 13, 119, 28/119, 76, 24/76, 89, 24/89, 70, 21/70, 55, 22/55, , 22:  
 40, 14, 48, 10/48, 41, 23/41, 121, 23/121, 42, 39/42, 15, 39/15, 93, 58/  
 93, 118, 35/118, 123, 35/123, 16, 35/16, 96, 35/96, 124, 40/  
 124, 17, 40/17, 43, 40/43, 131, 48/131, 64, 24/64, 90, 24/90, 65, 12/  
 65, 89, 26/89, 70, 21/70, 55, 22/55, , 22:  
 41, 121, 42, 39/42, 15, 39/15, 93, 58/93, 118, 35/118, 123, 35/123, 16, 35/  
 16, 96, 35/96, 124, 40/124, 17, 40/17, 43, 40/43, 131, 48/131, 64, 24/  
 64, 90, 24/90, 65, 12/65, 89, 26/89, 70, 21/70, 55, 22/55, , 22:  
 42, 15, 93, 58/93, 118, 35/118, 123, 35/123, 16, 35/16, 96, 35/96, 124, 40/  
 124, 17, 40/17, 43, 40/43, 131, 48/131, 64, 24/64, 90, 24/90, 65, 12/  
 65, 89, 26/89, 70, 21/70, 55, 22/55, , 22:  
 43, 131, 64, 24/64, 90, 24/90, 65, 12/65, 89, 26/89, 70, 21/70, 55, 22/  
 55, , 22:  
 44, 86, 66, 16/66, 2, 26/2, 108, 26/108, 59, 40/59, 51, 39/51, , 39:  
 45, 87, 101, 16/101, 110, 36/110, 49, 36/49, 18, 36/18, 102, 37/  
 102, 111, 45/111, 19, 45/19, 54, 44/54, , 41:  
 46, 20, 104, 24/104, 114, 40/114, 125, 40/125, 105, 40/105, 24, 39/  
 24, 84, 39/84, 106, 46/106, 116, 47/116, 126, 47/126, 23, 47/  
 23, 103, 47/103, 107, 50/107, 128, 50/128, 74, 49/74, 85, 49/  
 85, 6, 12/6, 30, 4/30, 31, 9/31, 109, 12/109, 1, 28/1, 56, 28/56, , 9:  
 47, 113, 135, 37/135, 46, 37/46, 20, 38/20, 104, 24/104, 114, 40/  
 114, 125, 40/125, 105, 40/105, 24, 39/24, 84, 39/84, 106, 46/  
 106, 116, 47/116, 126, 47/126, 23, 47/23, 103, 47/103, 107, 50/  
 107, 128, 50/128, 74, 49/74, 85, 49/85, 6, 12/6, 30, 4/30, 31, 9/  
 31, 109, 12/109, 1, 28/1, 56, 28/56, , 9:  
 48, 82, 115, 40/115, 83, 40/83, 136, 40/136, 22, 39/22, 98, 39/  
 98, 132, 44/132, 21, 45/21, 97, 45/97, 122, 38/122, 75, 38/  
 75, 90, 38/90, 65, 12/65, 89, 26/89, 70, 21/70, 55, 22/55, , 22:  
 49, 81, 137, 39/137, 40, 40/40, 14, 39/14, 48, 10/48, 41, 23/41, 121, 23/  
 121, 42, 39/42, 15, 39/15, 93, 58/93, 118, 35/118, 123, 35/  
 123, 16, 35/16, 96, 35/96, 124, 40/124, 17, 40/17, 43, 40/  
 43, 131, 48/131, 64, 24/64, 90, 24/90, 65, 12/65, 89, 26/89, 70, 21/  
 70, 55, 22/55, , 22:  
 50, 36, 10, 24/10, 11, 10/11, 12, 16/12, 69, 16/69, 37, 32/37, 95, 32/  
 95, 77, 44/77, 38, 44/38, 94, 44/94, 78, 45/78, 80, 45/80, 39, 45/  
 39, 13, 44/13, 119, 28/119, 76, 24/76, 89, 24/89, 70, 21/70, 55, 22/  
 55, , 22:  
 51, 25, 52, 22/52, , 50:  
 52, 138, 139, 18/139, 4, 18/4, 27, 18/27, 134, 21/134, 71, 22/71, 88, 22/  
 88, 5, 29/5, 67, 29/67, 28, 52/28, 29, 52/29, 63, 10/63, 85, 30/  
 85, 6, 21/6, 30, 4/30, 31, 9/31, 109, 12/109, 1, 28/1, 56, 28/56, , 9:  
 53, 140, 26, 26/26, 3, 26/3, 92, 21/92, 133, 36/133, 60, 38/60, 88, 38/  
 88, 5, 29/5, 67, 29/67, 28, 52/28, 29, 52/29, 63, 10/63, 85, 30/  
 85, 6, 21/6, 30, 4/30, 31, 9/31, 109, 12/109, 1, 28/1, 56, 28/56, , 9:  
 54, 99, 7, 52/7, 68, 36/68, 8, 42/8, 33, 42/33, 34, 14/34, 73, 2/73, 91, 30/  
 91, 47, 30/47, 35, 17/35, 53, 17/53, , 34:  
 55, 99, 32, 52/32, 9, 180/9, 62, 40:  
 56, 124, 17, 40/17, 43, 40/43, 131, 48/131, 64, 24/64, 90, 24/90, 65, 12/  
 65, 89, 26/89, 70, 21/70, 55, 22/55, , 22;

```

;
;   INITIALIZE ALL VARIABLES USED TO GATHER PERCENT LOADED,
;   UNLOADED TRAVEL, LOADING/UNLOADING, PARKING, AND WAITING
;   TIME
;
INITIALIZE,X(1)=0,X(2)=0,X(3)=0,X(4)=0;
;
;   PRINT OUT THE DATA COLLECTED DURING THE SIMULATION RUN.
;
TALLIES:1,TIME IN SYSTEM:2,TRAVEL LOADED:3,TRAVEL UNLOADED:
      4,WAITING TIME:5,WAITING TIME:6,PARKING TIME;
DSTAT:1,X(3),OUT OF PARKING:2,X(4),IN TO PARKING;
COUNTERS:1,THROUGHPUT;
;
;   SPECIFY THE NUMBER OF SEGMENTS IN THE SYSTEM AND THE
;   NUMBER AND LENGTH OF THE SIMULATION RUN(S).
;
RESOURCES:1-150,PATH;
REPLICA,1,0,28800;
END;

```



BEGIN,1,1,YES,thmr,NO;

THMREV.MOD  
(CASE 2 - IMPROVED LAYOUT)

GENERATION OF JOBS FOR EACH INPUT STATION. FROM THE  
GIVEN DATA, THE NUMBER OF JOBS AT EACH STATION IS KNOWN  
AND THE INTER-ARRIVAL TIME BETWEEN JOBS IS TAKEN TO BE  
UNIFORMLY DISTRIBUTED. THE FIRST JOB (AT EACH STATION)  
IS CREATED AT .5% OF ITS EXPECTED INTER-ARRIVAL TIME.  
THE JOBS IS GENERATED WITH RANDOM NUMBER GENERATOR  
STREAM # 2.

CREATE,1,1:UN(26,2);  
ASSIGN:A(4)=1;  
BRANCH,1:ALWAYS,DCIN;  
CREATE,1,2:UN(27,2);  
ASSIGN:A(4)=2;  
BRANCH,1:ALWAYS,DCIN;  
CREATE,1,6:UN(28,2);  
ASSIGN:A(4)=3;  
BRANCH,1:ALWAYS,DCIN;  
CREATE,1,12:UN(29,2);  
ASSIGN:A(4)=4;  
BRANCH,1:ALWAYS,DCIN;  
CREATE,1,18:UN(30,2);  
ASSIGN:A(4)=5;  
BRANCH,1:ALWAYS,DCIN;  
CREATE,1,5:UN(31,2);  
ASSIGN:A(4)=6;  
BRANCH,1:ALWAYS,DCIN;  
CREATE,1,103:UN(32,2);  
ASSIGN:A(4)=7;  
BRANCH,1:ALWAYS,DCIN;  
CREATE,1,69:UN(33,2);  
ASSIGN:A(4)=8;  
BRANCH,1:ALWAYS,DCIN;  
CREATE,1,69:UN(34,2);  
ASSIGN:A(4)=9;  
BRANCH,1:ALWAYS,DCIN;  
CREATE,1,18:UN(35,2);  
ASSIGN:A(4)=10;  
BRANCH,1:ALWAYS,DCIN;  
CREATE,1,6:UN(36,2);  
ASSIGN:A(4)=11;  
BRANCH,1:ALWAYS,DCIN;  
CREATE,1,2:UN(37,2);  
ASSIGN:A(4)=12;  
BRANCH,1:ALWAYS,DCIN;  
CREATE,1,3:UN(38,2);  
ASSIGN:A(4)=13;

```

        BRANCH,1:ALWAYS,DCIN;
        CREATE,1,7:UN(39,2);
        ASSIGN:A(4)=14;
        BRANCH,1:ALWAYS,DCIN;
        CREATE,1,9:UN(40,2);
        ASSIGN:A(4)=15;
        BRANCH,1:ALWAYS,DCIN;
        CREATE,1,6:UN(41,2);
        ASSIGN:A(4)=16;
        BRANCH,1:ALWAYS,DCIN;
        CREATE,1,6:UN(42,2);
        ASSIGN:A(4)=17;
        BRANCH,1:ALWAYS,DCIN;
        CREATE,1,2:UN(43,2);
        ASSIGN:A(4)=18;
        BRANCH,1:ALWAYS,DCIN;
        CREATE,1,2:UN(44,2);
        ASSIGN:A(4)=19;
        BRANCH,1:ALWAYS,DCIN;
        CREATE,1,1:UN(45,2);
        ASSIGN:A(4)=20;
        BRANCH,1:ALWAYS,DCIN;
        CREATE,1,2:UN(46,2);
        ASSIGN:A(4)=21;
        BRANCH,1:ALWAYS,DCIN;
        CREATE,1,2:UN(47,2);
        ASSIGN:A(4)=22;
        BRANCH,1:ALWAYS,DCIN;
        CREATE,1,2:UN(48,2);
        ASSIGN:A(4)=23;
        BRANCH,1:ALWAYS,DCIN;
        CREATE,1,2:UN(49,2);
        ASSIGN:A(4)=24;
;
;   LIMIT THE TOTAL CONCURRENT JOBS IN THE SYSTEM TO 250.
;
;
DCIN   ASSIGN:
        X(2)=X(2)+1;
        BRANCH,1:
            IF,X(2).GT.250,END;
            ELSE,REQ;
;
;   IF TOTAL CONCURRENT JOBS > 250, DISPOSE THE INCOMING
;   JOBS.
;
END    ASSIGN:
        X(2)=X(2)-1:DISPOSE;
;
;   REQUEST FOR AN AGV (SEND A SIGNAL). IF THERE IS AN
;   AVAILABLE AGV IN THE PARKING LOT, REQUEST THAT AGV.
;   OTHERWISE WAIT FOR THE NEXT AVAILABLE AGV.

```

```

;
REQ      SIGNAL:1:MARK(10);
        ASSIGN:M=A(4);
        ASSIGN:A(11)=38;
;
;      WAIT UNTIL AN AGV IS ALLOCATED.
;
;      QUEUE,M;
;      WAIT:M+50,1;
;      ASSIGN:A(11)=39;
;
;      IF AN AGV HAS BEEN ALLOCATED, WAIT UNTIL IT REACHES THE
;      JOB.
;
;      QUEUE,M+175;
;      WAIT:M+105,1;
;
;      GATHER STATISTICS FOR THE TIME BETWEEN JOB CREATION AND
;      THE TIME THE JOB IS PICKED UP, AND DECREMENT THE TOTAL
;      CONCURRENT JOBS IN THE SYSTEM.
;
;      ASSIGN:X(2)=X(2)-1;
;      TALLY:1,INT(10):DISPOSE;
;
;      GENERATION OF AGVs. ALL AGVs ARE CREATED AT THE
;      BEGINNING OF THE SIMULATION.
;
;      CREATE,1:0,21;
;      ASSIGN:X(1)=X(1)+1;
;      ASSIGN:A(12)=X(1);
;
;      WAIT FOR A REQUEST.
;
WAIT      ASSIGN:A(11)=40;
        QUEUE,25:MARK(7);
        WAIT:1,1;
;
;      IF THE SIGNAL FOR A REQUEST (JOB) IS ACCEPTED, FIND THE
;      JOB USING THE MAXIMUM QUEUE SIZE SELECTION RULE. IF
;      THE JOB IS FOUND, GET THE JOB. IF NOT, WAIT FOR ANOTHER
;      REQUEST.
;      NOTE: THIS BLOCK WILL BE SUBSTITUTED WITH A FORTRAN
;      SUBROUTINE (USING AN EVENT BLOCK) WHEN THE
;      SHORTEST DISTANCE ALGORITHM IS USED.
;
FIND      FINDJ,1,24:
        MAX(NQ(J));
        BRANCH,1:
        IF,NQ(J).EQ.0,WAIT:
        ELSE,OUT;
;

```

```

;   IF THE JOB IS FOUND, SEND A SIGNAL TO THE JOB TO INDICATE
;   THAT AN AGV HAS BEEN ALLOCATED.  SET ON EMPTY TRAVEL
;   TIME, AND GATHER STATISTICS FOR PARKING TIME.
;
OUT   ASSIGN:A(5)=J:MARK(14);
      TALLY:6,INT(7);
      ASSIGN:A(11)=37;
      SIGNAL:A(5)+50;
      ASSIGN:M=150;
;
;   IF THERE IS ANOTHER AGV GOING OUT OF PARKING LOT, WAIT
;   UNTIL THAT AGV GETS OUT OF PARKING LOT.
;
      QUEUE,175:MARK(15);
      SEIZE:PATH(150);
      TALLY:4,INT(15);
      ASSIGN:X(3)=1;
      DELAY:UN(50,1),M;
      ASSIGN:X(3)=0;
;
;   INITIALIZE THE ROUTE, ASSIGN SPEED AND CINEMA SYMBOL FOR
;   AN EMPTY AGV, AND SET ON EMPTY TRAVEL TIME.
;
      ASSIGN:NS=69;
      ASSIGN:A(1)=50;
      ASSIGN:A(9)=0;
      ASSIGN:IS=0;
      ASSIGN:A(8)=5.0;
      ASSIGN:A(6)=0;
      ASSIGN:A(11)=37;
      ROUTE:5,96;
;
;   STATION (MACRO) --
;   CHECK (1) IF THE AGV JUST GOT OUT OF PARKING LOT
;   (2) IF AN INTERSECTION IS REACHED.  IF IT'S,
;   ASSIGN THE ROUTE TO DESTINATION.
;
      STATION,1-149;
      BRANCH,1:
        IF,(M.EQ.96).AND.(A(9).EQ.0),
          OPKR:
            IF,(M.GE.47).AND.(M.LE.57),NRT:
              ELSE,NXT0;
NXT0  ASSIGN:A(3)=M;
NXT2  ASSIGN:M=A(1);
;
;   SET ON WAITING TIME, SEIZE THE NEXT SEGMENT.  IF THE NEXT
;   SEGMENT IS OCCUPIED, WAIT UNTIL IT'S FREED.  OTHERWISE,
;   SEIZE THE SEGMENT AND RELEASE THE OCCUPIED SEGMENT.
;
NBLK  QUEUE,M+25:MARK(15);

```

```

SEIZE:PATH(M);
TALLY:4,INT(15);
RELEASE:PATH(A(3));
ASSIGN:M=A(3);
;
; CHECK IF (1) THE AGV JUST GOT OUT OF PARKING LOT
; (2) THE AGV IS EMPTY (WITHOUT A REQUEST)
; (3) THE DESTINATION IS REACHED
; IF NOT (1), (2), OR (3), PROCEED TO NEXT STATION.
;
BRANCH,1:
IF,A(9).EQ.0,STA:
IF,A(6).EQ.3,CHK:
IF,A(5).EQ.A(3),ARR:
ELSE,GO;
STA ASSIGN:M=96;
ASSIGN:A(9)=1;
GO ROUTE:A(2)/A(8),SEQ;
;
; ARRIVAL -- COLLECT STATISTICS FOR EITHER EMPTY OR LOADED
; TRAVEL, SET ON UNLOADING/LOADING TIME, AND
; PICK UP OR DROP OFF THE JOB.
;
ARR BRANCH,1:
IF,A(6).EQ.1,I13:
ELSE,I14;
I13 TALLY:2,INT(13);
BRANCH,1:ALWAYS,DEL;
I14 TALLY:3,INT(14);
DEL DELAY:UN(25,1),M:MARK(16);
TALLY:5,INT(16);
ASSIGN:NS=A(5);
ASSIGN:IS=0;
BRANCH,1:
IF,A(6).EQ.1,DROP:
ELSE,DELV;
;
; DELIVERY -- SET ON LOADED TRAVEL TIME, AND ASSIGN THE
; DESTINATION OF THE JOB ACCORDING TO THE
; GIVEN PROBABILITY (FROM THE DATA).
;
DELV ASSIGN:A(8)=4.0:MARK(13);
ASSIGN:A(6)=1;
ASSIGN:A(4)=A(5);
ASSIGN:J=A(4)+105;
SIGNAL:J;
ASSIGN:A(5)=DP(A(4),1);
;
; ASSIGN CINEMA SYMBOL ACCORDING TO THE STATION OF ORIGIN
; AND THE DESTINATION OF THE JOB.
;

```

```

;
;   BRANCH,1:ALWAYS,CNM;
;
;   DROP-OFF -- SET ON EMPTY TRAVEL TIME, AND CHECK IF THERE
;               IS ANYMORE REQUEST.
;
DROP      ASSIGN:A(8)=5.0:MARK(14);
          ASSIGN:A(6)=0;
          ASSIGN:A(11)=37;
          COUNT:1;
;
;   FIND ANOTHER JOB. IF THERE IS NONE, SEND THE AGV TO THE
;   PARKING LOT AND CHECK FOR MORE JOB ON THE WAY TO PARKING
;   LOT. IF PARKING LOT IS REACHED, PARK THE AGV. IF NOT,
;   SEND THE AGV TO THE NEXT STATION.
;   NOTE: THE FOLLOWING BLOCKS WILL BE SUBSTITUTED WITH A
;   FORTRAN SUBROUTINE (USING AN EVENT BLOCK) WHEN
;   THE SHORTEST DISTANCE ALGORITHM IS USED TO FIND
;   THE NEXT JOB.
;
CHK      FINDJ,1,24:MAX(NQ(J));
          BRANCH,1:
            IF,(NQ(J).EQ.0).AND.(M.EQ.96),S150:
            IF,NQ(J).EQ.0,PKR:
            ELSE,PORD;
;
;   PARKING STATION -- WAIT IF THERE IS ANOTHER AGV IS
;   PARKING, AND SET OFF UNLOADED TRAVEL TIME.
;
S150     QUEUE,200:MARK(15);
          SEIZE:PATH(151);
          TALLY:4,INT(15);
          RELEASE:PATH(50);
          ASSIGN:M=96;
          ROUTE:5,151;
          STATION,151;
          RELEASE:PATH(151);
          ASSIGN:X(4)=1;
          DELAY:UN(50,1),M;
          ASSIGN:X(4)=0;
          TALLY:3,INT(14):MARK(7);
          BRANCH,1:ALWAYS,FIND;
;
;   IF ANOTHER JOB IS FOUND, PICK UP THE JOB.
;
PORD     ASSIGN:A(5)=J;
          SIGNAL:A(5)+50;
          ASSIGN:A(6)=0;
          BRANCH,1:
            IF,M.EQ.A(5),ARR:
            ELSE,GO;
;

```

```

;      IF NO MORE JOB IS FOUND, SEND THE AGV TO PARKING LOT.
;
PKR      ASSIGN:A(5)=50;
          ASSIGN:A(6)=3;
          ASSIGN:A(11)=39;
          ROUTE:A(2)/A(8),SEQ;
OPKR     ASSIGN:A(3)=150;
          BRANCH,1:ALWAYS,NXT2;
;
;      THE FOLLOWING BLOCKS ASSIGN THE SHORTEST ROUTE TO THE
;      DESTINATION (FROM AN INTERSECTION).
;      NOTE:  THESE BLOCKS WILL BE SUBSTITUTED WITH A FORTRAN
;             SUBROUTINE (USING AN EVENT BLOCK) WHEN THE
;             SHORTEST DISTANCE ALGORITHM IS USED.
;
NRT      ASSIGN:IS=0;
          ASSIGN:J=0;
          BRANCH,1:
            IF,(A(5).EQ.14).OR.(A(5).EQ.15),J1:
            IF,(A(5).GE.40).AND.(A(5).LE.42),J1:
            IF,(A(5).EQ.36).OR.(A(5).EQ.37),J2:
            IF,(A(5).GE.10).AND.(A(5).LE.12),J2:
            IF,(A(5).GE.33).AND.(A(5).LE.35),J3:
            IF,(A(5).EQ.7).OR.(A(5).EQ.8),J3:
            IF,(A(5).EQ.9).OR.(A(5).EQ.32),J4:
            IF,(A(5).EQ.5).OR.(A(5).EQ.28),J5:
            IF,A(5).EQ.29,J5:
            IF,(A(5).EQ.6).OR.(A(5).EQ.30),J6:
            IF,A(5).EQ.31,J6:
            IF,(A(5).EQ.20).OR.(A(5).EQ.46),J7:
            IF,(A(5).EQ.23).OR.(A(5).EQ.24),J7:
            IF,(A(5).EQ.18).OR.(A(5).EQ.19),J8:
            IF,(A(5).EQ.21).OR.(A(5).EQ.22),J9:
            ELSE,RDST;
J1       ASSIGN:J=1;
          BRANCH,1:ALWAYS,RDST;
J2       ASSIGN:J=2;
          BRANCH,1:ALWAYS,RDST;
J3       ASSIGN:J=3;
          BRANCH,1:ALWAYS,RDST;
J4       ASSIGN:J=4;
          BRANCH,1:ALWAYS,RDST;
J5       ASSIGN:J=5;
          BRANCH,1:ALWAYS,RDST;
J6       ASSIGN:J=6;
          BRANCH,1:ALWAYS,RDST;
J7       ASSIGN:J=7;
          BRANCH,1:ALWAYS,RDST;
J8       ASSIGN:J=8;
          BRANCH,1:ALWAYS,RDST;
J9       ASSIGN:J=9;

```

```

RDST      BRANCH,1:ALWAYS,RDST;
          BRANCH,1:
            IF,M.EQ.47,C47:
            IF,M.EQ.48,C48:
            IF,M.EQ.49,C49:
            IF,M.EQ.50,C50:
            IF,M.EQ.51,C51:
            IF,M.EQ.52,C52:
            IF,M.EQ.53,C53:
            IF,M.EQ.54,C54:
            IF,M.EQ.55,C55:
            IF,M.EQ.56,C55:
            IF,M.EQ.57,C57;
C47        ASSIGN:NS=65;
          ASSIGN:A(1)=129;
          BRANCH,1:
            IF,(J.GE.7).AND.(J.LE.9),J47A:
            IF,(A(5).EQ.2).OR.(A(5).EQ.13),J47A:
            ELSE,NXT0;
J47A       ASSIGN:NS=64;
          ASSIGN:A(1)=13;
          BRANCH,1:ALWAYS,NXT0;
C48        ASSIGN:NS=58;
          ASSIGN:A(1)=115;
          BRANCH,1:
            IF,(J.EQ.5).OR.(A(5).EQ.27),J48A:
            IF,(A(5).EQ.6).OR.(A(5).EQ.30),J48A:
            IF,A(5).EQ.31,J48B:
            ELSE,NXT0;
J48A       ASSIGN:NS=59;
          ASSIGN:A(1)=27;
          BRANCH,1:ALWAYS,NXT0;
J48B       ASSIGN:NS=57;
          BRANCH,1:ALWAYS,NXT0;
C49        ASSIGN:NS=60;
          ASSIGN:A(1)=78;
          BRANCH,1:
            IF,(A(5).EQ.16).OR.(A(5).EQ.150),J49A:
            IF,(A(5).EQ.17).OR.(A(5).EQ.43),J49A:
            ELSE,NXT0;
J49A       ASSIGN:NS=61;
          ASSIGN:A(1)=145;
          BRANCH,1:ALWAYS,NXT0;
C50        ASSIGN:NS=63;
          ASSIGN:A(1)=100;
          BRANCH,1:
            IF,(J.GE.7).AND.(J.LE.9),J50A:
            IF,(A(5).EQ.17).OR.(A(5).EQ.43),J50A:
            IF,A(5).EQ.2,J50A:
            ELSE,NXT0;
J50A       ASSIGN:NS=62;

```



```

C51      ASSIGN:A(1)=124;
          BRANCH,1:ALWAYS,NXT0;
          ASSIGN:NS=51;
          ASSIGN:A(1)=58;
          BRANCH,1:
            IF,(J.GE.5).AND.(J.LE.9),J51A:
            IF,(A(5).EQ.1).OR.(A(5).EQ.2),J51A:
            IF,(A(5).EQ.4).OR.(A(5).EQ.27),J51A:
            ELSE,NXT0;
J51A     ASSIGN:NS=52;
          ASSIGN:A(1)=138;
          BRANCH,1:ALWAYS,NXT0;
C52      ASSIGN:NS=56;
          ASSIGN:A(1)=99;
          BRANCH,1:
            IF,(J.GE.5).AND.(J.LE.9),J52A:
            IF,(A(5).EQ.3).OR.(A(5).EQ.26),J52A:
            IF,(J.EQ.4),J52B:
            IF,(J.GE.1).AND.(J.LE.3),J52C:
            ELSE,NXT0;
J52A     ASSIGN:NS=53;
          ASSIGN:A(1)=140;
          BRANCH,1:ALWAYS,NXT0;
J52B     ASSIGN:NS=55;
          BRANCH,1:ALWAYS,NXT0;
J52C     ASSIGN:NS=54;
          BRANCH,1:ALWAYS,NXT0;
C53      ASSIGN:NS=50;
          ASSIGN:A(1)=36;
          BRANCH,1:
            IF,(A(5).GE.14).AND.(A(5).LE.17),J53A:
            IF,(A(5).GE.40).AND.(A(5).LE.43),J53A:
            IF,A(5).EQ.150,J53A:
            ELSE,NXT0;
J53A     ASSIGN:NS=49;
          ASSIGN:A(1)=81;
          BRANCH,1:ALWAYS,NXT0;
C54      ASSIGN:NS=48;
          ASSIGN:A(1)=82;
          BRANCH,1:
            IF,(J.EQ.6).OR.(J.EQ.7),J54A:
            IF,A(5).EQ.1,J54A:
            ELSE,NXT0;
J54A     ASSIGN:NS=47;
          ASSIGN:A(1)=113;
          BRANCH,1:ALWAYS,NXT0;
C55      ASSIGN:NS=44;
          ASSIGN:A(1)=86;
          BRANCH,1:
            IF,(J.GE.7).AND.(J.LE.9),J55A:
            IF,(A(5).EQ.6).OR.(A(5).EQ.30),J55A:

```

```

ELSE, NXT0;
J55A  ASSIGN: NS=45;
      ASSIGN: A(1)=87;
      BRANCH, 1: ALWAYS, NXT0;
C57   ASSIGN: NS=66;
      ASSIGN: A(1)=148;
      BRANCH, 1:
        IF, (A(5).EQ.19).OR.(A(5).EQ.22), J57A:
        IF, (A(5).EQ.46).OR.(A(5).EQ.20), J57A:
        IF, A(5).EQ.24, J57A:
        IF, J.EQ.6, J57B:
        IF, (A(5).EQ.1).OR.(A(5).EQ.23), J57B:
        ELSE, NXT0;
J57A  ASSIGN: NS=68;
      ASSIGN: A(1)=111;
      BRANCH, 1: ALWAYS, NXT0;
J57B  ASSIGN: NS=67;
      ASSIGN: A(1)=144;
      BRANCH, 1: ALWAYS, NXT0;
;
;     THE FOLLOWING BLOCKS ASSIGN THE CINEMA SYMBOL ACCORDING
;     TO THE STATION ORIGIN AND THE DESTINATION OF THE JOB.
;     NOTE: THESE BLOCKS WILL BE SUBSTITUTED WITH A FORTRAN
;           SUBROUTINE (AN EVENT BLOCK) WHEN THE SHORTEST
;           DISTANCE ALGORITHM IS USED.
;
CNM   BRANCH, 1:
      IF, (A(4).GE.14).AND.(A(4).LE.17), ONE:
      IF, (A(4).GE.10).AND.(A(4).LE.13), TWO:
      IF, (A(4).EQ.3).OR.(A(4).EQ.4), THR:
      IF, (A(4).GE.7).AND.(A(4).LE.9), THR:
      IF, (A(4).EQ.1).OR.(A(4).EQ.2), FOUR:
      IF, (A(4).EQ.5).OR.(A(4).EQ.6), FOUR:
      IF, (A(4).EQ.21).OR.(A(4).EQ.22), FIVE:
      IF, (A(4).EQ.18).OR.(A(4).EQ.19), FIVE:
      ELSE, SIX;
ONE   ASSIGN: J=0;
      BRANCH, 1: ALWAYS, DEST;
TWO   ASSIGN: J=1;
      BRANCH, 1: ALWAYS, DEST;
THR   ASSIGN: J=2;
      BRANCH, 1: ALWAYS, DEST;
FOUR  ASSIGN: J=3;
      BRANCH, 1: ALWAYS, DEST;
FIVE  ASSIGN: J=4;
      BRANCH, 1: ALWAYS, DEST;
SIX   ASSIGN: J=5;
      BRANCH, 1: ALWAYS, DEST;
DEST  BRANCH, 1:
      IF, (A(5).GE.40).AND.(A(5).LE.43), E11:
      IF, (A(5).GE.36).AND.(A(5).LE.39), E12:

```

```

      IF, (A(5).GE.25).AND.(A(5).LE.27), E13:
      IF, (A(5).GE.32).AND.(A(5).LE.35), E13:
      IF, (A(5).GE.28).AND.(A(5).LE.31), E14:
      IF, (A(5).EQ.18).OR.(A(5).EQ.19), E15:
      IF, (A(5).EQ.21).OR.(A(5).EQ.22), E15:
      ELSE, E16;
E11    ASSIGN:A(11)=J*6+1;
      ROUTE:A(2)/A(8), SEQ;
E12    ASSIGN:A(11)=J*6+2;
      ROUTE:A(2)/A(8), SEQ;
E13    ASSIGN:A(11)=J*6+3;
      ROUTE:A(2)/A(8), SEQ;
E14    ASSIGN:A(11)=J*6+4;
      ROUTE:A(2)/A(8), SEQ;
E15    ASSIGN:A(11)=J*6+5;
      ROUTE:A(2)/A(8), SEQ;
E16    ASSIGN:A(11)=J*6+6;
      ROUTE:A(2)/A(8), SEQ;
END;

```

```

      SUBROUTINE EVENT(JOB,I)
C*****
C      STHMREV.FOR
C      (CASE 2 - IMPROVED LAYOUT)
C
C      THIS SUBROUTINE DETERMINES WHICH EVENT OR WHICH ONE OF
C      THE FOLLOWING THREE SUBROUTINES WILL BE EXECUTED
C      THESE SUBROUTINES ARE USED FOR THE SHORTEST DISTANCE
C      ALGORITHM.
C*****
C      GOTO (1,2,3),I
C      1 CALL CINDEF(JOB)
C        RETURN
C      2 CALL RSDEF(JOB)
C        RETURN
C      3 CALL SHORSTDST(JOB)
C        END
C
C
C      SUBROUTINE CINDEF(JOB)
C*****
C      THIS SUBROUTINE ASSIGN THE CINEMA SYMBOL, ACCORDING TO
C      THE STATION ORIGIN AND THE DESTINATION OF THE JOB, TO
C      ATTRIBUTE (11) OF THE JOB.
C
C      ORIG = STATION ORIGIN (STORED IN ATTRIBUTE # 4)
C      DEST = DESTINATION (STORED IN ATTRIBUTE # 5)
C      CATT = ZONE FOR THE STATION ORIGIN
C      DATT = ZONE FOR THE DESTINATION
C      VAL = THE CINEMA SYMBOL
C*****
C      COMMON/SIM/D(50),DL(50),S(50),SL(50),X(50),DTNOW,TNOW,
C      1TFIN,J,NRUN
C      ORIG=A(JOB,4)
C      DEST=A(JOB,5)
C      IF ((ORIG.GE.14).AND.(ORIG.LE.17)) THEN
C        CATT=0.0
C      ELSEIF ((ORIG.GE.10).AND.(ORIG.LE.13)) THEN
C        CATT=1.0
C      ELSEIF ((ORIG.EQ.3).OR.(ORIG.EQ.4)) THEN
C        CATT=2.0
C      ELSEIF ((ORIG.GE.7).AND.(ORIG.LE.9)) THEN
C        CATT=2.0
C      ELSEIF ((ORIG.EQ.1).OR.(ORIG.EQ.2)) THEN
C        CATT=3.0
C      ELSEIF ((ORIG.EQ.5).OR.(ORIG.EQ.6)) THEN

```

```

      CATT=3.0
    ELSEIF ((ORIG.EQ.21).OR.(ORIG.EQ.22)) THEN
      CATT=4.0
    ELSEIF ((ORIG.EQ.18).OR.(ORIG.EQ.19)) THEN
      CATT=4.0
    ELSE
      CATT=5.0
    ENDIF
    IF ((DEST.GE.40).AND.(DEST.LE.43)) THEN
      VAL=(CATT*6.0)+1.0
    ELSEIF ((DEST.GE.36).AND.(DEST.LE.39)) THEN
      VAL=(CATT*6.0)+2.0
    ELSEIF ((DEST.GE.25).AND.(DEST.LE.27)) THEN
      VAL=(CATT*6.0)+3.0
    ELSEIF ((DEST.GE.32).AND.(DEST.LE.35)) THEN
      VAL=(CATT*6.0)+3.0
    ELSEIF ((DEST.GE.28).AND.(DEST.LE.31)) THEN
      VAL=(CATT*6.0)+4.0
    ELSEIF ((DEST.EQ.18).OR.(DEST.EQ.19)) THEN
      VAL=(CATT*6.0)+5.0
    ELSEIF ((DEST.EQ.21).OR.(DEST.EQ.22)) THEN
      VAL=(CATT*6.0)+5.0
    ELSE
      VAL=(CATT*6.0)+6.0
    ENDIF
    CALL SETA(JOB,11,VAL)
    RETURN
  END
C
C
  SUBROUTINE RSDEF(JOB)
C*****
C
C   THIS SUBROUTINE ASSIGN THE SHORTEST ROUTE THE AGV NEEDS
C   TO TAKE TO GET TO ITS DESTINATION AT EACH INTERSECTION.
C
C   IJUNCT  =  INTERSECTION NUMBER (CURRENT LOCATION OF
C               THE AGV)
C   DEST    =  FINAL DESTINATION OF THE AGV
C   INUM    =  SEQUENCE (ROUTE) NUMBER
C   VAL     =  NEXT STATION TO BE SEIZE
C   JCT     =  ZONE FOR THE STATIONS
C
C*****
C
  COMMON/SIM/D(50),DL(50),S(50),SL(50),X(50),DTNOW,TNOW,
  1TFIN,J,NRUN
  IJUNCT=M(JOB)
  DEST=A(JOB,5)
  JCT=0
  NSECT=0

```

```

IF ((DEST.EQ.14).OR.(DEST.EQ.15)) THEN
  NSECT=1
ELSEIF ((DEST.GE.40).AND.(DEST.LE.42)) THEN
  NSECT=1
ELSEIF ((DEST.EQ.36).OR.(DEST.EQ.37)) THEN
  NSECT=2
ELSEIF ((DEST.GE.10).AND.(DEST.LE.12)) THEN
  NSECT=2
ELSEIF ((DEST.GE.33).AND.(DEST.LE.35)) THEN
  NSECT=3
ELSEIF ((DEST.EQ.7).OR.(DEST.EQ.8)) THEN
  NSECT=3
ELSEIF ((DEST.EQ.9).OR.(DEST.EQ.32)) THEN
  NSECT=4
ELSEIF ((DEST.EQ.5).OR.(DEST.EQ.28)) THEN
  NSECT=5
ELSEIF (DEST.EQ.29) THEN
  NSECT=5
ELSEIF ((DEST.EQ.6).OR.(DEST.EQ.30)) THEN
  NSECT=6
ELSEIF (DEST.EQ.31) THEN
  NSECT=6
ELSEIF ((DEST.EQ.20).OR.(DEST.EQ.46)) THEN
  NSECT=7
ELSEIF ((DEST.EQ.23).OR.(DEST.EQ.24)) THEN
  NSECT=7
ELSEIF ((DEST.EQ.18).OR.(DEST.EQ.19)) THEN
  NSECT=8
ELSEIF ((DEST.EQ.21).OR.(DEST.EQ.22)) THEN
  NSECT=9
ENDIF
IF (IJUNCT.EQ.47) THEN
  IF ((NSECT.GE.7).AND.(NSECT.LE.9)) THEN
    INUM=64
    VAL=13
  ELSEIF ((DEST.EQ.2).OR.(DEST.EQ.13)) THEN
    INUM=64
    VAL=13
  ELSE
    INUM=65
    VAL=129
  ENDIF
ELSEIF (IJUNCT.EQ.48) THEN
  IF ((NSECT.EQ.5).OR.(DEST.EQ.27)) THEN
    INUM=59
    VAL=27
  ELSEIF ((DEST.EQ.6).OR.(DEST.EQ.30)) THEN
    INUM=59
    VAL=27
  ELSEIF (DEST.EQ.31) THEN
    INUM=57
  ENDIF

```

```

        VAL=115
    ELSE
        INUM=58
        VAL=115
    ENDIF
ELSEIF (IJUNCT.EQ.49) THEN
    IF ((DEST.EQ.16).OR.(DEST.EQ.150)) THEN
        INUM=61
        VAL=145
    ELSEIF ((DEST.EQ.43).OR.(DEST.EQ.17)) THEN
        INUM=61
        VAL=145
    ELSE
        INUM=60
        VAL=78
    ENDIF
ELSEIF (IJUNCT.EQ.50) THEN
    VAL=124
    IF ((NSECT.GE.7).AND.(NSECT.LE.9)) THEN
        INUM=62
    ELSEIF ((DEST.EQ.17).OR.(DEST.EQ.43)) THEN
        INUM=62
    ELSEIF (DEST.EQ.2) THEN
        INUM=62
    ELSE
        INUM=63
        VAL=100
    ENDIF
ELSEIF (IJUNCT.EQ.51) THEN
    VAL=138
    IF ((NSECT.GE.5).AND.(NSECT.LE.9)) THEN
        INUM=52
    ELSEIF ((DEST.EQ.1).OR.(DEST.EQ.2)) THEN
        INUM=52
    ELSEIF ((DEST.EQ.4).OR.(DEST.EQ.27)) THEN
        INUM=52
    ELSE
        INUM=51
        VAL=58
    ENDIF
ELSEIF (IJUNCT.EQ.52) THEN
    VAL=99
    IF ((NSECT.GE.5).AND.(NSECT.LE.9)) THEN
        INUM=53
        VAL=140
    ELSEIF ((DEST.EQ.3).OR.(DEST.EQ.26)) THEN
        INUM=53
        VAL=140
    ELSEIF (NSECT.EQ.4) THEN
        INUM=55
    ELSEIF ((NSECT.GE.1).AND.(NSECT.LE.3)) THEN

```

```

        INUM=54
    ELSE
        INUM=56
    ENDIF
ELSEIF (IJUNCT.EQ.53) THEN
    VAL=81
    IF ((DEST.GE.40).AND.(DEST.LE.43)) THEN
        INUM=49
    ELSEIF ((DEST.GE.14).AND.(DEST.LE.17)) THEN
        INUM=49
    ELSEIF (DEST.EQ.150) THEN
        INUM=49
    ELSE
        INUM=50
        VAL=36
    ENDIF
ELSEIF (IJUNCT.EQ.54) THEN
    IF ((NSECT.EQ.6).OR.(NSECT.EQ.7)) THEN
        INUM=47
        VAL=113
    ELSEIF (DEST.EQ.1) THEN
        INUM=47
        VAL=113
    ELSE
        INUM=48
        VAL=82
    ENDIF
ELSEIF ((IJUNCT.EQ.55).OR.(IJUNCT.EQ.56)) THEN
    IF ((NSECT.GE.7).AND.(NSECT.LE.9)) THEN
        INUM=45
        VAL=87
    ELSEIF ((DEST.EQ.6).OR.(DEST.EQ.30)) THEN
        INUM=45
        VAL=87
    ELSE
        INUM=44
        VAL=86
    ENDIF
ELSEIF (IJUNCT.EQ.57) THEN
    VAL=111
    IF ((DEST.EQ.19).OR.(DEST.EQ.22)) THEN
        INUM=68
    ELSEIF ((DEST.EQ.46).OR.(DEST.EQ.20)) THEN
        INUM=68
    ELSEIF (DEST.EQ.24) THEN
        INUM=68
    ELSEIF (NSECT.EQ.6) THEN
        INUM=67
        VAL=144
    ELSEIF ((DEST.EQ.1).OR.(DEST.EQ.23)) THEN
        INUM=67

```



```

        VAL=144
    ELSE
        INUM=66
        VAL=148
    ENDIF
ENDIF
CALL SETA(JOB,5,DEST)
CALL SETNS(JOB,INUM)
CALL SETIS(JOB,0)
CALL SETA(JOB,1,VAL)
RETURN
END

C
C
SUBROUTINE SHORTDST(JOB)
C*****
C
C    THIS SUBROUTINE WILL FIND THE CLOSEST REQUEST TO THE AGV.
C    THIS SUBROUTINE READS DATA FORM FILE "THMREV.DAT",
C    WHICH CONTAINS THE SEQUENCE OF INPUT STATIONS.
C
C    Z(K)      =  AN ARRAY THAT STORE THE INPUT STATION NUMBERS
C    TJOB      =  STATUS OF THE AGV:
C                  0 - PICKING UP A JOB
C                  3 - EMPTY (WITHOUT A REQUEST)
C
C    LASTDST   =  CURRENT LOCATION OF THE AGV
C    DEST      =  FINAL DESTINATION OF THE AGV
C    IONE      =  DROP-OFF STATION NUMBER
C    ITWO      =  ZERO, A CHECK TO SEE IF THE LINE READ (IN
C                  THMREV.DAT) IS CORRECT.
C*****
C
COMMON/SIM/D(50),DL(50),S(50),SL(50),X(50),DTNOW,TNOW,
1TFIN,J,NRUN
DIMENSION Z(24)
OPEN(10,FILE='THMREV.DAT')
REWIND 10
LASTDST=A(JOB,4)
33 READ(10,*) IONE,ITWO
IF ((IONE.EQ.LASTDST).AND.(ITWO.EQ.0)) THEN
    READ(10,*) Z(1),Z(2),Z(3),Z(4),Z(5),Z(6),Z(7),Z(8)
    READ(10,*) Z(9),Z(10),Z(11),Z(12),Z(13),Z(14),Z(15),Z(16)
    READ(10,*) Z(17),Z(18),Z(19),Z(20),Z(21),Z(22),Z(23),Z(24)
ELSE
    READ(10,*) IONE,ITWO
    READ(10,*) IONE,ITWO
    READ(10,*) IONE,ITWO
    GOTO 33
ENDIF
C

```

C CHECK IF THERE IS ANY REQUEST AT THE CLOSEST INPUT  
C STATION. IF THERE IS NONE, CHECK AT THE NEXT CLOSEST  
C INPUT STATIONS. IF THERE IS NO MORE REQUEST, SEND THE  
C AGV TO THE PARKING LOT (STATION 100).  
C

```
    DEST=150
    TJOB=0
    DO 1 K=1,24
        N=Z(K)
        IF ((NQ(N).NE.0).AND.(DEST.EQ.150)) DEST=N
1  CONTINUE
    IF (DEST.EQ.150) TJOB=3
    CALL SETA(JOB,5,DEST)
    CALL SETA(JOB,6,TJOB)
    RETURN
    END
```

\*\*\*\*\*

THMREV.DAT  
(CASE 2 - IMPROVED LAYOUT)

THIS FILE CONTAINS THE DATA TO FIND THE CLOSEST  
STATION WITH REQUEST. THE ORGANIZATION OF THE DATA IS AS  
FOLLOWS: LINE 1 : DROP-OFF OR INTERSECTION STATION, ZERO  
LINE 2 : THE FIRST EIGHT CLOSEST INPUT STATIONS  
LINE 3 : THE SECOND EIGHT CLOSEST INPUT STATIONS  
LINE 4 : THE LAST EIGHT CLOSEST INPUT STATIONS

\*\*\*\*\*

18 0  
18 23 19 21 20 22 6 1  
24 2 4 3 7 5 8 9  
16 10 11 13 12 14 17 15  
19 0  
19 20 22 21 24 23 2 18  
4 6 1 3 7 5 8 9  
16 10 11 13 12 14 17 15  
20 0  
20 22 21 24 23 2 18 4  
6 1 3 7 5 8 19 9  
16 10 11 13 12 14 17 15  
21 0  
21 2 18 4 19 23 3 7  
5 8 1 20 6 9 22 16  
10 11 13 12 24 14 17 15  
22 0  
22 21 2 18 4 19 23 3  
7 5 8 1 20 6 9 16  
10 11 13 12 24 14 17 15  
23 0  
23 6 1 2 18 4 19 3  
7 5 21 8 20 9 22 16  
10 11 13 12 24 14 17 15  
24 0  
24 23 6 1 2 18 4 19  
3 7 5 21 8 20 9 22  
16 10 11 13 12 14 17 15  
25 0  
3 7 8 5 9 16 10 11  
12 13 6 14 17 1 2 4  
18 15 19 23 21 20 22 24  
26 0  
3 5 6 1 2 18 4 19  
23 7 21 8 20 9 22 16  
10 11 13 12 24 14 17 15  
27 0  
5 6 1 2 18 4 19 23  
3 7 21 8 20 9 22 16

10 11 13 12 24 14 17 15  
 28 0  
 6 1 2 18 4 19 23 3  
 7 5 21 8 20 9 22 16  
 10 11 13 12 24 14 17 15  
 29 0  
 6 1 2 18 4 19 23 3  
 7 5 21 8 20 9 22 16  
 10 11 13 12 24 14 17 15  
 30 0  
 1 2 18 4 19 23 3 7  
 5 21 8 20 6 9 22 16  
 10 11 13 12 24 14 17 15  
 31 0  
 1 2 18 4 19 23 3 7  
 5 21 8 20 6 9 22 16  
 10 11 13 12 24 14 17 15  
 32 0  
 9 10 11 12 16 13 17 2  
 4 18 3 7 5 1 8 19  
 23 6 21 20 22 14 24 15  
 33 0  
 10 11 12 16 13 17 2 4  
 18 3 7 5 1 8 19 23  
 6 9 21 20 22 14 24 15  
 34 0  
 10 11 12 16 13 17 2 4  
 18 3 7 5 1 8 19 23  
 6 9 21 20 22 14 24 15  
 35 0  
 10 11 12 16 13 17 2 4  
 18 3 7 5 1 8 19 23  
 6 9 21 20 22 14 24 15  
 36 0  
 10 11 12 16 13 17 2 4  
 18 3 7 5 8 19 23 1  
 6 21 9 20 22 14 24 15  
 37 0  
 16 13 17 2 4 18 3 7  
 5 8 19 23 1 6 21 9  
 10 11 12 20 22 14 24 15  
 38 0  
 16 13 17 2 4 18 3 7  
 5 8 19 23 1 6 21 9  
 10 11 12 20 22 14 24 15  
 39 0  
 13 2 4 18 3 7 5 8  
 19 23 1 6 9 21 16 10  
 11 12 20 22 14 17 15 24  
 40 0  
 14 15 16 17 13 2 4 18

3 7 5 8 19 23 1 6  
 9 21 10 11 12 20 22 24  
 41 0  
 15 16 17 13 2 4 18 3  
 7 5 8 19 23 1 6 9  
 21 10 11 12 20 22 14 24  
 42 0  
 15 17 13 2 4 18 3 7  
 5 8 19 23 1 6 9 21  
 16 10 11 12 20 22 14 24  
 43 0  
 2 18 4 19 23 3 7 5  
 21 8 1 20 6 9 22 16  
 10 11 13 12 24 14 17 15  
 46 0  
 20 22 21 24 23 2 18 4  
 6 1 3 7 5 8 19 9  
 16 10 11 13 12 14 17 15  
 47 0  
 13 2 4 18 3 7 5 8  
 19 23 1 6 9 21 16 10  
 11 12 20 22 14 17 15 24  
 48 0  
 5 1 6 2 18 4 19 23  
 3 7 21 8 20 9 22 16  
 10 11 13 12 24 14 17 15  
 49 0  
 16 13 17 2 4 18 3 7  
 5 8 19 23 1 6 21 9  
 10 11 12 20 22 14 24 15  
 50 0  
 17 13 2 4 18 3 7 5  
 8 19 23 1 6 9 21 16  
 10 11 12 20 22 14 24 15  
 51 0  
 4 3 7 5 8 1 2 6  
 9 16 10 11 12 13 18 14  
 17 19 23 15 21 20 22 24  
 52 0  
 3 7 8 5 9 16 10 11  
 12 13 6 14 17 1 2 4  
 18 15 19 23 21 20 22 24  
 53 0  
 10 11 12 16 13 17 2 4  
 18 3 7 5 1 8 19 23  
 6 9 21 20 22 14 24 15  
 54 0  
 20 22 21 24 23 2 18 4  
 6 1 3 7 5 8 19 9  
 16 10 11 13 12 14 17 15  
 55 0

2 18 4 19 23 3 7 5  
21 8 1 20 6 9 22 16  
10 11 13 12 24 14 17 15  
56 0  
2 18 4 19 23 3 7 5  
21 8 1 20 6 9 22 16  
10 11 13 12 24 14 17 15  
57 0  
23 19 21 20 22 6 1 24  
2 18 4 3 7 5 8 9  
16 10 11 13 12 14 17 15

```

BEGIN;
;
;
;           THMREV.EXP
;       ( CASE 2 - IMPROVED LAYOUT)
;
;       GIVE THE TITLE OF THE SUMMARY GENERATED AT THE END OF
;       SIMULATION RUN.
;
PROJECT,CASE 2 REVISED Q D,I. OETOMO,12/20/87;

;       DEFINE THE MAX. NUMBER OF CONCURRENT ENTITIES,
;       ATTRIBUTES, QUEUES, AND THE ATTRIBUTE NUMBER FOR CINEMA
;       SYMBOL.
;
DISCRETE,300,16,200,152,11;
;
;       INITIALIZE THE SEED VALUE FOR THE RANDOM NUMBER GENERATOR
;       STREAM (TWO SETS OF SEED VALUES: 3000, 5555 & 7777, 1000,
;       ARE USED TO SIMULATE EACH NUMBER OF AGVs), AND DEFINE ALL
;       PARAMETERS VALUES USED IN THE MODEL FILE.
;
SEEDS:1,7777:2,1000;
PARAMETERS:
1,.014,28,.495,29,.569,18,.643,19,.704,46,.778,21,.852,22,
.926,23,1.,24:
2,.216,27,.34,18,.465,19,.502,46,.626,21,.751,22,.875,23,
1.,24:
3,.631,43,.684,18,.736,19,.789,46,.842,21,.894,22,.947,23,
1.,24:
4,.5,38,1.,39:
5,.073,30,.207,18,.341,19,.463,46,.597,21,.731,22,.865,23,
1.,24:
6,.459,29,.466,43,.543,18,.62,19,.691,46,.768,21,.845,22,
.922,23,1,24:
7,.142,18,.285,19,.428,46,.571,21,.714,22,.857,23,1.,24:
8,.142,18,.285,19,.428,46,.571,21,.714,22,.857,23,1.,24:
9,.142,18,.285,19,.428,46,.571,21,.714,22,.857,23,1.,24:
10,.641,26,.692,18,.743,19,.794,46,.846,21,.897,22,.948,23,
1.,24:
11,.819,43,.844,18,.868,19,.901,46,.926,21,.948,22,.974,23,
1.,24:
12,.821,43,.844,18,.868,19,.901,46,.926,21,.950,22,.975,23,
1.,24:
13,.063,27,.898,31,.915,18,.932,19,.949,21,.966,22,.983,23,
1.,24:
14,.707,43,.75,18,.792,19,.830,46,.872,27,.915,22,.957,23,
1.,24:
15,1.,43:16,1.,43:17,1.,43:
18,.087,25,.158,26,.170,28,.175,29,.203,30,.333,31,.347,32,
.361,33,.368,34,.447,35,.462,36,.493,37,.672,38,.85,39,
.892,40,.921,41,.953,42,1.,46:

```

```

19,.087,25,.158,26,.170,28,.175,29,.203,30,.333,31,.347,32,
   .361,33,.368,34,.447,35,.462,36,.493,37,.672,38,.85,39,
   .892,40,.921,41,.953,42,1.,46:
20,.017,25,.038,26,.039,29,.059,30,.189,31,.193,32,.197,33,
   .199,34,.222,35,.226,36,.235,37,.260,38,.285,39,.298,40,
   .307,41,.316,42,.430,18,.544,19,.658,21,.772,22,.886,23,
   1.,24:
21,.087,25,.158,26,.170,28,.175,29,.203,30,.333,31,.347,32,
   .361,33,.368,34,.447,35,.462,36,.493,37,.672,38,.85,39,
   .892,40,.921,41,.953,42,1.,46:
22,.087,25,.158,26,.170,28,.175,29,.203,30,.333,31,.347,32,
   .361,33,.368,34,.447,35,.462,36,.493,37,.672,38,.85,39,
   .892,40,.921,41,.953,42,1.,46:
23,.087,25,.158,26,.170,28,.175,29,.203,30,.333,31,.347,32,
   .361,33,.368,34,.447,35,.462,36,.493,37,.672,38,.85,39,
   .892,40,.921,41,.953,42,1.,46:
24,.087,25,.158,26,.170,28,.175,29,.203,30,.333,31,.347,32,
   .361,33,.368,34,.447,35,.462,36,.493,37,.672,38,.85,39,
   .892,40,.921,41,.953,42,1.,46:
25,60,90:
26,202,247:27,355,433:28,1137,1389:29,2160,2640:30,3161,3863:
31,870,1063:32,18514,22629:33,12343,15086:34,12323,15086:
35,3323,4062:36,1062,1298:37,292,356:38,549,671:39,1223,1494:
40,1620,1980:41,1037,1267:42,1037,1267:43,392,479:44,392,479:
45,259,317:46,392,479:47,392,479:48,392,479:49,392,479:50,30,50;
;
;   DEFINE THE ROUTE FROM EACH INPUT, DROP-OFF STATIONS, AND
;   INTERSECTIONS (SEE EXPLANATION IN OHM.EXP FILE).
;
SEQUENCES:
1,56,,9:
2,108,59,32/59,51,46/51,,45:
3,92,133,36/133,60,38/60,88,38/88,5,29/5,102,29/102,67,30/
   67,28,37/28,29,52/29,63,10/63,85,30/85,6,21/6,30,4/30,31,9/
   31,109,12/109,1,28/1,56,28/56,,9:
4,48,,9:
5,102,67,30/67,28,37/28,29,52/29,63,10/63,85,30/85,6,21/
   6,30,4/30,31,9/31,109,12/109,1,28/1,56,28/56,,9:
6,30,31,9/31,109,12/109,1,28/1,56,28/56,,9:
7,68,8,42/8,33,42/33,34,14/34,73,2/73,91,30/91,35,30/
   35,53,34/53,,34:
8,33,34,14/34,73,2/73,91,30/91,35,30/35,53,34/53,,34:
9,62,91,90/91,35,30/35,53,34/53,,34:
10,11,12,16/12,69,16/69,37,32/37,95,32/95,77,35/77,120,40/
   120,38,30/38,94,27/94,49,36/49,,36:
11,12,69,16/69,37,32/37,95,32/95,77,35/77,120,40/120,38,30/
   38,94,27/94,49,36/49,,36:
12,69,37,32/37,95,32/95,77,35/77,120,40/120,38,30/38,94,27/
   94,49,36/49,,36:
13,119,76,24/76,89,24/89,70,21/70,55,22/55,,22:
14,44,41,23/41,121,23/121,42,39/42,15,39/15,93,58/93,118,35/

```



118, 123, 35/123, 16, 35/16, 96, 35/96, 50, 40/50, , 28:  
 15, 93, 118, 35/118, 123, 35/123, 16, 35/16, 96, 35/96, 50, 40/50, , 28:  
 16, 96, 50, 40/50, , 28:  
 17, 43, 131, 48/131, 64, 24/64, 90, 24/90, 65, 12/65, 89, 26/89, 70, 21/  
 70, 55, 22/55, , 22:  
 18, 57, , 45:  
 19, 54, , 41:  
 20, 104, 114, 40/114, 125, 40/125, 105, 40/105, 24, 39/24, 84, 39/  
 84, 106, 46/106, 116, 47/116, 126, 47/126, 23, 47/23, 103, 47/  
 103, 107, 50/107, 128, 50/128, 74, 49/74, 85, 49/85, 6, 12/  
 6, 30, 4/30, 31, 9/31, 109, 12/109, 1, 28/1, 56, 28/56, , 9:  
 21, 97, 122, 38/122, 75, 38/75, 90, 38/90, 65, 12/65, 89, 26/89, 70, 21/  
 70, 55, 22/55, , 22:  
 22, 98, 132, 44/132, 21, 45/21, 97, 45/97, 122, 38/122, 75, 38/  
 75, 90, 38/90, 65, 12/65, 89, 26/89, 70, 21/70, 55, 22/55, , 22:  
 23, 103, 107, 50/107, 128, 50/128, 74, 49/74, 85, 49/85, 6, 12/6, 30, 4/  
 30, 31, 9/31, 109, 12/109, 1, 28/1, 56, 28/56, , 9:  
 24, 84, 106, 46/106, 116, 47/116, 126, 47/126, 23, 47/23, 103, 47/  
 103, 107, 50/107, 128, 50/128, 74, 49/74, 85, 49/85, 6, 12/6, 30, 4/  
 30, 31, 9/31, 109, 12/109, 1, 28/1, 56, 28/56, , 9:  
 25, 52, , 50:  
 26, 3, 92, 21/92, 133, 36/133, 60, 38/60, 88, 38/88, 5, 29/5, 102, 29/  
 102, 67, 30/67, 28, 37/28, 29, 52/29, 63, 10/63, 85, 30/85, 6, 21/  
 6, 30, 4/30, 31, 9/31, 109, 12/109, 1, 28/1, 56, 28/56, , 9:  
 27, 134, 71, 22/71, 88, 22/88, 5, 29/5, 102, 29/102, 67, 30/67, 28, 37/  
 28, 29, 52/29, 63, 10/63, 85, 30/85, 6, 21/6, 30, 4/30, 31, 9/  
 31, 109, 12/109, 1, 28/1, 56, 28/56, , 9:  
 28, 29, 63, 10/63, 85, 30/85, 6, 21/6, 30, 4/30, 31, 9/31, 109, 12/  
 109, 1, 28/1, 56, 28/56, , 9:  
 29, 63, 85, 30/85, 6, 21/6, 30, 4/30, 31, 9/31, 109, 12/109, 1, 28/  
 1, 56, 28/56, , 9:  
 30, 31, 109, 12/109, 1, 28/1, 56, 28/56, , 9:  
 31, 109, 1, 28/1, 56, 28/56, , 9:  
 32, 9, 62, 40/62, 91, 90/91, 35, 30/35, 53, 34/53, , 34:  
 33, 34, 73, 2/73, 91, 30/91, 35, 30/35, 53, 34/53, , 34:  
 34, 73, 91, 30/91, 35, 30/35, 53, 34/53, , 34:  
 35, 53, , 34:  
 36, 10, 11, 10/11, 12, 16/12, 69, 16/69, 37, 32/37, 95, 32/95, 77, 35/  
 77, 120, 40/120, 38, 30/38, 94, 27/94, 49, 36/49, , 36:  
 37, 95, 77, 35/77, 120, 40/120, 38, 30/38, 94, 27/94, 49, 36/49, , 36:  
 38, 94, 49, 36/49, , 36:  
 39, 47, , 0:  
 40, 14, 44, 10/44, 41, 23/41, 121, 23/121, 42, 39/42, 15, 39/15, 93, 58/  
 93, 118, 35/118, 123, 35/123, 16, 35/16, 96, 35/96, 50, 40/50, , 28:  
 41, 121, 42, 39/42, 15, 39/15, 93, 58/93, 118, 35/118, 123, 35/  
 123, 16, 35/16, 96, 35/96, 50, 40/50, , 28:  
 42, 15, 93, 58/93, 118, 35/118, 123, 35/123, 16, 35/16, 96, 35/  
 96, 50, 40/50, , 28:  
 43, 131, 64, 24/64, 90, 24/90, 65, 12/65, 89, 26/89, 70, 21/  
 70, 55, 22/55, , 22:  
 44, 86, 66, 16/66, 2, 26/2, 108, 26/108, 59, 12/59, 51, 46/51, , 45:

45, 87, 101, 16/101, 110, 36/110, 139, 36/139, 18, 36/18, 57, 37/57, , 45:  
 46, 20, 104, 24/104, 114, 40/114, 125, 40/125, 105, 40/105, 24, 39/  
 24, 84, 39/84, 106, 46/106, 116, 47/116, 126, 47/126, 23, 47/  
 23, 103, 47/103, 107, 50/107, 128, 50/128, 74, 49/74, 85, 49/  
 85, 6, 12/6, 30, 4/30, 31, 9/31, 109, 12/109, 1, 28/1, 56, 28/56, , 9:  
 47, 113, 135, 37/135, 46, 37/46, 20, 38/20, 104, 24/104, 114, 40/  
 114, 125, 40/125, 105, 40/105, 24, 39/24, 84, 39/84, 106, 46/  
 106, 116, 47/116, 126, 47/126, 23, 47/23, 103, 47/103, 107, 50/  
 107, 128, 50/128, 74, 49/74, 85, 49/85, 6, 12/6, 30, 4/30, 31, 9/  
 31, 109, 12/109, 1, 28/1, 56, 28/56, , 9:  
 48, 82, 79, 40/79, 83, 40/83, 136, 40/136, 22, 39/22, 98, 39/98, 132, 44/  
 132, 21, 45/21, 97, 45/97, 122, 38/122, 75, 38/75, 90, 38/90, 65, 12/  
 65, 89, 26/89, 70, 21/70, 55, 22/55, , 22:  
 49, 81, 137, 39/137, 40, 40/40, 14, 39/14, 44, 10/44, 41, 23/41, 121, 23/  
 121, 42, 39/42, 15, 39/15, 93, 58/93, 118, 35/118, 123, 35/  
 123, 16, 35/16, 96, 35/96, 50, 40/50, , 28:  
 50, 36, 10, 24/10, 11, 10/11, 12, 16/12, 69, 16/69, 37, 32/37, 95, 32/  
 95, 77, 35/77, 120, 40/120, 38, 30/38, 94, 27/94, 49, 36/49, , 36:  
 51, 58, 25, 19/25, 52, 18/52, , 50:  
 52, 138, 45, 23/45, 4, 23/4, 48, 23/48, , 9:  
 53, 140, 26, 26/26, 3, 26/3, 92, 21/92, 133, 36/133, 60, 38/60, 88, 38/  
 88, 5, 29/5, 102, 29/102, 67, 30/67, 28, 37/28, 29, 52/29, 63, 10/  
 63, 85, 30/85, 6, 21/6, 30, 4/30, 31, 9/31, 109, 12/109, 1, 28/  
 1, 56, 28/56, , 9:  
 54, 99, 7, 52/7, 68, 36/68, 8, 42/8, 33, 42/33, 34, 14/34, 73, 2/  
 73, 91, 30/91, 35, 30/35, 53, 34/53, , 34:  
 55, 99, 32, 52/32, 9, 180/9, 62, 40:  
 56, 99, 112, 52/112, 117, 26/117, 120, 26/120, 38, 25/38, 94, 27/  
 94, 49, 36/49, , 36:  
 57, 115, 142, 38/142, 143, 40/143, 146, 40/146, 31, 40/31, 109, 40/  
 109, 1, 28/1, 56, 28/56, , 8:  
 58, 115, 142, 38/142, 143, 40/143, 147, 40/147, 109, 34/109, 1, 34/  
 1, 56, 28/56, , 8:  
 59, 27, 134, 12/134, 71, 22/71, 88, 22/88, 5, 29/5, 102, 29/102, 67, 30/  
 67, 28, 37/28, 29, 52/29, 63, 10/63, 85, 30/85, 6, 21/6, 30, 4/  
 30, 31, 9/31, 109, 12/109, 1, 28/1, 56, 28/56, , 9:  
 60, 78, 80, 40/80, 39, 40/39, 47, 27/47, , 0:  
 61, 145, 72, 39/72, 16, 39/16, 96, 38/96, 50, 40/50, , 28:  
 62, 124, 17, 26/17, 43, 26/43, 131, 48/131, 64, 24/64, 90, 24/90, 65, 12/  
 65, 89, 26/89, 70, 21/70, 55, 22/55, , 22:  
 63, 100, 149, 31/149, 127, 31/127, 39, 31/39, 47, 27/47, , 0:  
 64, 13, 119, 28/119, 76, 24/76, 89, 24/89, 70, 21/70, 55, 22/55, , 22:  
 65, 129, 130, 31/130, 59, 31/59, 51, 46/51, , 45:  
 66, 148, 141, 38/141, 98, 38/98, 132, 38/132, 21, 45/21, 97, 45/  
 97, 122, 38/122, 75, 38/75, 90, 38/90, 65, 12/65, 89, 26/89, 70, 21/  
 70, 55, 22/55, , 22:  
 67, 144, 61, 38/61, 23, 38/23, 103, 38/103, 107, 50/107, 128, 50/  
 128, 74, 49/74, 85, 49/85, 6, 12/6, 30, 4/30, 31, 9/31, 109, 12/  
 109, 1, 28/1, 56, 28/56, , 9:  
 68, 111, 19, 45/19, 54, 44/54, , 41:  
 69, 50, , 28;

```

;
;   INITIALIZE ALL VARIABLES USED TO GATHER PERCENT LOADED,
;   UNLOADED TRAVEL, LOADING/UNLOADING, PARKING, AND WAITING
;   TIME
;
INITIALIZE,X(1)=0,X(2)=0,X(3)=0,X(4)=0;
;
;   PRINT OUT THE DATA COLLECTED DURING THE SIMULATION RUN.
;
TALLIES:1,TIME IN SYSTEM:2,TRAVEL LOADED:3,TRAVEL UNLOADED:
        4,WAITING TIME:5,LOADING TIME:6,PARKING TIME;
DSTAT:1,X(3),OUT OF PARKING:2,X(4),IN TO PARKING;
COUNTERS:1,THROUGHPUT;
;
;   SPECIFY THE NUMBER OF SEGMENTS IN THE SYSTEM AND THE
;   NUMBER AND LENGTH OF THE SIMULATION RUN(S).
;
RESOURCES:1-152,PATH;
REPLICA,1,0,28800;
END;

```

AN INVESTIGATION OF TRAFFIC CONGESTION IN AN AUTOMATED GUIDED  
VEHICLE MATERIAL HANDLING SYSTEM USING ANIMATION (CINEMA/EGA)

by

INGKO OETOMO

B.S., KANSAS STATE UNIVERSITY, 1986

-----

AN ABSTRACT OF A MASTER'S THESIS

submitted in partial fulfillment of the

requirements for the degree

MASTER OF SCIENCE

Department of Industrial Engineering

KANSAS STATE UNIVERSITY  
Manhattan, Kansas

1988

## ABSTRACT

This thesis investigates traffic congestion in an Automated Guided Vehicle (AGV) Material Handling System using animation (CINEMA/EGA).

Two selection algorithms: shortest distance and maximum queue size were used in the proposed AGV systems. The AGVS were evaluated using the animation (CINEMA/EGA) and the statistics gathered (SIMAN). The shortest distance algorithm transported more parts/shift than the maximum queue size algorithm. The average, minimum and maximum parts waiting time for the shortest distance algorithm were smaller than those of the maximum queue size algorithm.

Animation was found to be an effective tool to investigate traffic congestion. It could also be used to verify and debug our model or programming logic.

CINEMA animation could be readily constructed when the model was small to medium size. However, as the model becomes large, the animation becomes more complicated. The size and orientation of the station and queue symbols made the animation construction rather difficult.